Survivale Network Design Problems with High Connectivity Requirement

Ibrahima Diarrassouba

N° d'ordre : D.U. 1989
EDSPIC : 467

Université Blaise Pascal - Clermont II

École Doctorale
Sciences Pour l'Ingénieur de Clermont-Ferrand

THÈSE

présentée par

Ibrahima DIARRASSOUBA

pour obtenir le grade de

Docteur d'université
Spécialité : Informatique

Survivable Network Design Problems with High Connectivity Requirement

Soutenue publiquement le 07 Décembre 2009 devant le jury :

A.	Quilliot	Président du jury
J.-F.	Maurras	Rapporteur
G.	Oriolo	Rapporteur
H.	Yaman	Rapporteur
M.	Didi Biha	Examinateur
J.	Mailfert	Examinateur
M.	Haouari	Invité
A.R.	Mahjoub	Directeur de thèse

A ma mère, mon père, ma sœur Aliman
et toute la famille Diarrassouba.

Remerciements

Je tiens à remercier Mr A. Ridha Mahjoub, Professeur à l'Université Paris Dauphine de Paris, de m'avoir permis d'effectuer cette thèse sous sa direction. Je lui suis profondément reconnaissant pour la confiance qu'il m'a accordé en me permettant d'effectuer mon stage de D.E.A. puis une thèse sous sa direction. Je le remercie pour sa constante disponibilité, ses conseils et son soutien dans les moments difficiles. Grâce à lui, j'ai découvert le monde passionnant de la recherche et de l'enseignement ainsi que la rigueur scientifique qu'ils imposent. Je lui en serai toujours reconnaissant.

J'ai été très honoré que Mr Jean-François Maurras, Professeur à l'Université de la Méditérranée de Marseille, ait accepté de rapporter ma thèse. Je lui exprime pour cela mes plus sincères remerciements et pour les commentaires qu'il a apporté.

Je remercie Mr Gianpaolo Oriolo, Professeur à l'Université de Rome Tor Vergata (Italie), pour avoir accepté de rapporter ma thèse et pour sa lecture approfondie de ce manuscrit ainsi que pour l'intérêt qu'il a porté à mes travaux.

Je remercie également Mme Hande Yaman, Professeur à l'Université de Bilkent, Ankara (Turquie), de m'avoir fait l'honneur d'être rapporteur sur ma thèse et pour la lecture rapide et précise qu'elle a effectuée.

Je voudrais également remercier Mr Mohamed Haouari, Professeur à l'Ecole Polytechnique de Tunisie, pour l'intérêt qu'il a bien voulu porter à ma thèse et d'avoir accepté de participer à mon jury.

Mes remerciements vont également à Mr Alain Quilliot, Professeur à l'Université Blaise Pascal de Clermont-Ferrand, pour avoir accepté de présider le jury de cette thèse.

Je remercie aussi Mrs Mohamed Didi Biha et Jean Mailfert, respectivement Professeur à l'Université de Caen et Maître de Conférences à l'Université d'Auvergne de Clermont-Ferrand, d'avoir bien voulu examiner mes travaux et participer au jury.

Ces remerciements seraient incomplets sans une mention particulière pour Fatiha Bendali-Mailfert, Maître de Conférences à l'Université Blaise Pascal de Clermont-Ferrand, qui, avec Jean Mailfert, a participé à l'encadrement de mon stage de D.E.A. et d'une partie de ma thèse. Aussi, je tiens à les remercier tous les deux pour leur écoute et les précieux conseils qu'ils m'ont prodigués, tant sur le plan de la recherche et de l'enseignement que sur le plan personnel. Je remercie à nouveau Mohamed Didi Biha pour l'intérêt qu'il a porté à mon travail pendant toute la durée de ma thèse et son agréable contact.

Je tiens également à remercier tous les autres membres de l'équipe EPOC (Equipe Polyêdre et Optimisation Combinatoire) du laboratoire LIMOS de Clermont-Ferrand, au sein de laquelle j'ai passé des années agréables. Merci donc à Sylvie Borne pour sa bonne humeur, sa gentillesse et les discussions intéressantes que nous avons pu avoir. Merci à Denis Cornaz et Mathieu Lacroix pour les discussions scientifiques que nous avions dans notre bureau. Merci à Pierre Fouilhoux, David Huygens et Pierre Pesneau que j'ai eu plaisir à cotoyer au sein de l'équipe. Je remercie également Hervé Kerivin avec qui j'ai beaucoup apprécié de collaborer pour les enseignements. J'ai particulièrement apprécié son sérieux et sa rigueur et aussi les qualités scientifiques que je lui connais. J'adresse aussi mes plus sincères remerciements à Lise Slama avec qui j'ai aimé travailler et qui a toujours été pour moi une amie. Enfin, je remercie Rawiya Taktak pour son amitié et son soutien, ainsi que Sebastien Martin. Je leur souhaite à tous deux beaucoup de courage dans leurs thèses respectives.

Mes remerciements vont aussi à toute la famille Diarrassouba qui, malgré la distance, s'est toujours intéressée à mon travail. Je leur dédie à tous cette thèse. Je dédie aussi particulièrement à ma mère, qui est certainement fière de l'aboutissement de ce travail, et à mon père et ma soeur Aliman qui ne sont malheureusement plus avec nous et qui auraient été certainement fièrs de voir la fin de toutes ces années d'études. Je remercie également mon épouse Awa pour sa patience et ses encouragements et qui, durant toutes ses années, a su rester présente même dans les moments difficiles. J'adresse aussi un grand merci à Diaby Moussa AbdoulKader, son épouse Adjara et ses deux adorables filles Fatim et Khadija qui m'ont hébergé durant mon année d'ATER à Paris et grâce à qui j'ai vécu des moments inoubliables.

Enfin, je termine ces remerciements en adressant ma plus profonde reconnaissance à tous mes amis de Clermont-Ferrand, Aïssata Coulibaly, Souleymane Diarra, Alassane Drabo, Françoise Lavadoux et tous les autres, pour leur amitié et leur soutien permanent pendant et surtout à la fin de ma thèse. Je leur souhaite beaucoup de courage dans leurs études respectives.

Résumé

Cette thèse s'inscrit dans le cadre d'une étude polyhédrale des problèmes de conception de réseaux fiables avec forte connexité. En particulier, nous considérons les problèmes dits du sous-graphe k-arête-connexe et de conception de réseau k-arête-connexe avec contrainte de borne lorsque $k \geq 3$.

Dans un premier temps, nous étudions le problème du sous-graphe k-arête-connexe. Etant donné un graphe non orienté et valué $G = (V, E)$ et un entier positif k, le problème du sous-graphe k-arête-connexe consiste à déterminer un sous-graphe de G de poids minimum telle qu'il existe k chaînes arête-disjointes entre chaque paire de sommets de V. Nous discutons du polytope associé à ce problème lorsque $k \geq 3$. Nous introduisons une nouvelle famille d'inégalités valides pour le polytope et présentons plusieurs familles d'inégalités valides. Pour chaque famille d'inégalités, nous étudions les conditions sous lesquelles ces inégalités définissent des facettes. Nous discutons aussi du problème de séparation associé à chaque famille d'inégalités ainsi que d'opérations de réduction de graphes. En utilisant ces résultats, nous développons un algorithme de coupes et branchements pour le problème et donnons des résultats exprérimentaux.

Ensuite, nous étudions le problème de conception de réseaux k-arête-connexe avec contrainte de borne. Soient $G = (V, E)$ un graphe valué non orienté, un ensemble de demandes $D \subseteq V \times V$ et deux entiers positifs k et L. Le problème de conception de réseaux k-arête-connexe avec contrainte de borne consiste à déterminer un sous-graphe de G de poids minimum telle qu'entre chaque paire de sommets $\{s, t\} \in D$, il existe k chaînes arête-disjointes de longueur au plus L. Nous étudions ce problème dans le cas où $k \geq 2$ et $L \in \{2, 3\}$. Nous examinons la structure du polytope associé et montrons que, lorsque $|D| = 1$, ce polytope est complètement décrit par les inégalités dites de st-coupe et de L-chemin-coupe avec les inégalités triviales. Ce résultat généralise ceux de Huygens et al. [75] pour $k = 2$, $L \in \{2, 3\}$ et Dahl et al. [35] pour $k \geq 2$, $L = 2$.

Enfin, nous nous intéressons au problème de conception de réseau k-arête-connexe avec contrainte de borne lorsque $k \geq 2$, $L \in \{2, 3\}$ et $|D| \geq 2$. Le problème est

NP-difficile dans ce cas. Nous introduisons quatre nouvelles formulations du problème sous la forme de programmes linéaires en nombres entiers. Celles-ci sont basées sur la transformation du graphe G en graphes orientés appropriés. Nous discutons du polytope associé à chaque formulation et introduisons plusieurs familles d'inégalités valides. Pour chacune d'elles, nous décrivons des conditions pour que ces inégalités définissent des facettes. En utilisant ces résultats, nous développons des algorithmes de coupes et branchements et de coupes, generation de colonnes et branchements pour le problème. Nous donnons des résultats expérimentaux et menons une étude comparative entre les différentes formulations.

Mots clés: Réseau fiable, graphe k-arête-connexe, chaîne de longueur bornée, polytope, facette, séparation, génération de colonnes, algorithme de coupes et branchements.

Abstract

This thesis presents a polyhedral study of survivable network design problems with high connectivity requirement. In particular, the k-edge-connected subgraph and the k-edge-connected hop-constrained network design problems when $k \geq 3$ are investigated.

We first consider the k-edge-connected subgraph problem. Given a weighted undirected graph $G = (V, E)$ and a positive integer k, the k-edge-connected subgraph problem is to find a minimum weight subgraph of G which contains k-edge-disjoint paths between every pair of nodes of V. We discuss the polytope associated with that problem when $k \geq 3$. We introduce a new class of valid inequalities and present several other classes of valid inequalities. For each class we study the conditions under which the concerned inequalities are facet defining. We also discuss the separation problem associated with each class of inequalities and consider some graph reduction operations. Using these results, we devise a Branch-and-Cut algorithm for the problem and give some computational results.

We also study the k-edge-connected hop-constrained network design problem. Let $G = (V, E)$ be a weighted undirected graph, a demand set $D \subseteq V \times V$, two positive integers k and L. The k-edge-connected hop-constrained network design problem is to find a minimum weight subgraph of G such that for every $\{s, t\} \in D$ there exist at least k-edge-disjoint st-paths of length at most L. We investigate the structure of the associated polytope when $k \geq 2$ and $L \in \{2, 3\}$. We show that, in the case where $|D| = 1$, this polytope is completely described by the so-called st-cut and L-path-cut inequalities toghether with the trivial inequalities. This result generalizes those obtained by Huygens et al. [75] for $k = 2$, $L \in \{2, 3\}$ and Dahl et al. [35] for $k \geq 2$, $L = 2$. We show that this complete description yields a polynomial time algorithm for the problem when $|D| = 1$, $k \geq 2$ and $L \in \{2, 3\}$.

We finally consider the k-edge-connected hop-constrained network design problem when $k \geq 2$, $L = 2, 3$ and $|D| \geq 2$. The problem is NP-hard in this case. We introduce four new integer programming formulations based on the transformation of

the graph G into appropriate directed graphs. We discuss the polytope associated with each formulation and introduce several classes of inequalities that are valid for these polytopes. We also study conditions for these inequalities to be facet defining. Using these results, we devise Branch-and-Cut and Branch-and-Cut-and-Price algorithms for the problem. We provide some computational results and a comparative study between the different formulations we have introduced for the problem.

Keywords: Survivable network, k-edge-connected graph, hop-constrained path, polytope, facet, separation, column generation, Branch-and-Cut algorithm.

Contents

List of Figures

Introduction

Telecommunications have a major importance in the functioning of modern societies. They are particularly important as many transactions are done throughout telecommunication networks. The appearance of fiber optic technology in telecommunications (1984) and the introduction of new generation network protocols (SONET/SDH, ATM, IP, MPLS, GMPLS, etc.) have allowed networks to convey more and more data. As a consequence, more complex applications such as video conference, Virtual Private Networks (VPN) and mobile telephony, have been developed and are used in various domains including finance, economy, medicine, scientific research and schooling.

Such an importance implies to have robust networks. Whatever the nature of a network, it must survive after any equipment network failure. In case of an outage of a network, the loss of money could reach several millions of euros. Survivable networks must satisfy some connectivity requirements that is, there exist a certain number of disjoint paths between some pair of nodes of the network. This condition ensures that the traffic can still be routed between two nodes after the failure of a given number of links or nodes, and that the network is still functional. One of the main objectives when designing a telecommunication network is to provide a sufficient degree of survivability, and this, with a minimum cost of construction and maintainance. Also, the dimensionning problem is often considered, that is to give the appropriate capacities to the links of the network in order to convey the traffic between some nodes and satisfy a given quality of service.

A network can be represented by a graph $G = (V, E)$ where V is the set of nodes and E, the set of edges. Different topologies have been proposed to design survivable networks. Each topology depends on the use of the network. However, as pointed out in [83] (see also [80]), the topology that seems to be very efficient (and needed in practice) is the uniform topology, that is to say that corresponding to networks that survive after the failures of $k - 1$ or fewer links, for some $k \geq 2$. The 2-connected topology ($k = 2$) provides an adequate level of survivability since most failure usually can be repaired relatively quickly. However, for many applications, a higher level of

connectivity may be necessary.

Another reliability condition concerns the length of the paths used to route the traffic. In fact, the alternative paths could be too long to guarantee an effective routing. In data networks, such as Internet, the elongation of the route of the information could cause a strong loss in the transfert speed and decrease the quality of service. For other networks, the signal itself could be degraded by a longer routing. In such cases, the L-path requirement (paths of length at most L), with $L \geq 2$, guarantees exactly the needed quality of the alternative routes.

Network design problems, as well as many combinatorial optimization problems, have been studied using different methods. Among those methods, the polyhedral approach has appeared to be very effective in solving difficult problems. This method, introduced by Edmonds [45], consists in reducing the resolution of a combinatorial optimization problem to that of a linear program. This is done thanks to the complete (or even partial) description of the polyhedron associated with the problem. The polyhedral approach is part of the exact methods used to solve combinatorial optimization problems.

The survivable network design problem has been widely studied when the connectivity requirement is low ($k = 2$). However, the high connectivity requirement case ($k \geq 3$) has received a little attention. In this thesis, we study the survivable network design problem with high connectivity requirement. In particular, we focus on two variants of the problem: when k-edge-disjoint paths are required between every pair of nodes (the k-edge-connected subgraph problem) and when k-edge-disjoint paths of length at most L are required between certain pairs of nodes (the k-edge-connected hop-constrained network design problem). The study is led using the polyhedral approach and provides exact and efficient algorithms to solve these problems.

This thesis is organized as follows. In Chapter 1, we present the basic notions and notations that will be used throughout this thesis. We also present a state-of-the-art on survivable network design problems. Chapters 2 and 3 deal with the k-edge-connected subgraph problem when $k \geq 3$. We study the polytope associated with this problem and devise a Branch-and-Cut algorithm. Chapters 4, 5 and 6 are dedicated to the k-edge-connected hop-constrained network design problem. In Chapter 4, we give a complete description of the polytope associated with the problem in the case where k-edge-disjoint L-paths are required between a single pair of nodes. We present a polynomial time cutting plane algorithm to solve the problem in this case. Chapters 5 and 6 concern the general case where the k-edge-disjoint L-paths are required between more than one pair of nodes of the network. We introduce new integer programming

formulations for this more general problem and study the associated polytopes. We devise Branch-and-Cut and Branch-and-Cut-and-Price algorithms for the problem and present extensive computational results.

Chapter 1

Preliminary Notions and State-of-the-Art

In this chapter we give some basic notions of combinatorial optimization, complexity theory and polyhedra. We present cutting plane and column generation methods as well as Branch-and-Cut and Branch-and-Cut-and-Price algorithms. We also present the basic definitions of graph theory that will be used throughout this thesis. Finally we give a state-of-the-art on the survivable network design problem.

1.1 Preliminary notions

1.1.1 Combinatorial optimization

Combinatorial Optimization is a branch of operations research and is related to computer science and applied mathematics. It aims to study optimization problems where the set of feasible solutions is discrete or can be represented as a discrete one. A combinatorial optimization problem can be formulated in the following way. Let $E = \{e_1, ..., e_n\}$ be a finite set called *basic set* where each element e_i is associated with a weight $w(e_i)$. Let $\mathcal{F}$ be a family of subset of E. If $F \in \mathcal{F}$, then $w(F) = \sum_{e_i \in F} w(e_i)$ is the weight of F. The problem consists in finding an element F^* of $\mathcal{F}$ whose weight is

minimum (or maximum).

$$\begin{cases} \text{Minimize (or Maximize)} w(F) \\ s.t. \\ F \in \mathcal{F}. \end{cases}$$

$\mathcal{F}$ is the set of feasible solutions of the problem. The term *optimization* means that we are looking for the best possible solution. The term *combinatorial* refers to the discrete structure of $\mathcal{F}$. Most of the time, this structure is represented by a graph. Also, the number of feasible solutions is generally exponential, which makes diffuclt or even impossible to solve a combinatorial optimization problem with an enumerative procedure. Different methods exist in the litterature to solve combinatorial optimization problems, especially graph theory, linear and non-linear programming, integer programming and polyhedral approach.

Many real-world problems can be formulated as combinatorial optimization ones such as the Knapsack Problem, the Travelling Salesman Problem, telecommunication network design problems, VLSI circuit design problems, machine sequencing problem, etc. Some of them are directly applied in everyday life. For example Video On Demand services (VOD) are studied as a combinatorial optimization problem. The objective is to satisfy the demand of every client (the end users) and such that the total bandwidth allocated by the telecommunication operator for the service is minimum. This way, the operator can evaluate the quality of the service he provides and the corresponding cost. Another example is the GPS (GPS stands for Global Positioning System) which helps a driver to find the best way (in terms of distance or in terms of time) to go from one place to another. This is a direct application of the shortest path problem.

Combinatorial optimization is closely related to algorithm theory and computational complexity theory. The next section introduces computational issues of combinatorial optimization.

1.1.2 Computational and complexity theory

Computational and complexity theory is a branch of computer science whose objective is to classify problems according to their inherent difficulty. We distinguish "easy" and "difficult" problems. Computational and complexity theory is based on the works of Cook [22], Edmonds [44] and Karp [77]. For more details on this topic, the reader is referred to [56].

A *problem* is a question whose answer is unknown and depends on some input parameters. A problem is specified by describing its input parameters and the property that these parameter must satisfy. An *instance* of a problem is obtained by giving a specific value to all its input parameters. A *resolution algorithm* is a procedure, that is a succession of elementary operations, which produces a solution for a given instance of the problem. The number of input parameters necessary to describe an instance of a problem is the *size* of that problem.

An algorithm is said to be *polynomial* when the number of elementary operations necessary to solve an instance of size n is bounded by a polynomial function in n. A problem is of *class P* if there exists a polynomial algorithm to solve it. We also say that this problem is *easy* or can be solved *quickly*.

A *decision problem* is a problem whose answer is either "yes" of "no". Let $\mathcal{P}$ be a decision problem and $\mathcal{I}$ the set of instances of that problem for which the answer is "yes". $\mathcal{P}$ is said to be of *class NP* (where NP stands for Nondeterministic Polynomial) if there exists a polynomial algorithm which can verify that the answer is "yes" for every instance of $\mathcal{I}$. Clearly, every problem of class P is also of class NP (see Figure 1.1).

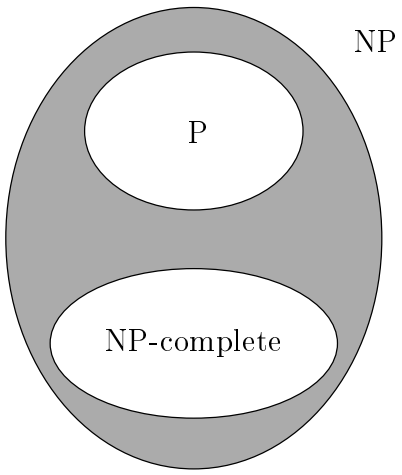


Figure 1.1: Relation between P, NP, NP-complete problems.

It is not known whether every problem in NP can be solved in polynomial time but it has been conjectured that $P = NP$. If this conjecture is proved, its consequence will be that every problem known as "difficult" can, in fact, be solved in polynomial time.

In the class NP, we distinguish a particular set of problems, the NP-complete problems. The notion of NP-completeness relies on the notion of polynomial reduction or transformation. A decision problem $\mathcal{P}_1$ can be polynomialy reduced (or transformed)

into another decision problem $\mathcal{P}_2$ if there exists a polynomial function f such that for every instance I of $\mathcal{P}_1$, the answer is "yes" if and only if the answer of $f(I)$ for $\mathcal{P}_2$ is "yes". A problem $\mathcal{P}$ is *NP-complete* if every problem of class NP can be polynomialy reduced into $\mathcal{P}$. The 3-satisfiability problem is the first problem showen to be NP-complete (see [22]).

Every combinatorial optimization problem can be associated with a decision problem. A combinatorial optimization problem whose decision problem is NP-complete is said to be *NP-hard*. Most of the combinatorial optimization problems are NP-hard. Among the methods used to solve them, the polyhedral approach has appeared to be very efficient.

1.1.3 Polyhedral approach and Branch-and-Cut method

Polyhedral theory has been introduced by Edmonds in 1965 [45]. He first developed this method for the matching problem. Later, further works were done on this topic. Polyhedral approach has appeared to be effective for solving many problems and slowly becomes a must for the study of combinatorial optimization problems. Here we present the basic notions of polyhedral theory. For more details, the reader is referred to [90, 96]. We also present the applied aspect of polyhedra to combinatorial optimization problems and describe the so-called Branch-and-Cut method.

1.1.3.1 Polyhedral theory

Let $n \in \mathbb{N}$ be a positive integer and $x \in \mathbb{R}^n$. We say that x is a *linear combination* of $x_1, ..., x_m \in \mathbb{R}^n$ if there exist m scalar $\lambda_1, ..., \lambda_m$ such that $x = \sum_{i=1}^{m} \lambda_i x_i$. If $\sum_{i=1}^{m} \lambda_i = 1$, then x is said to be an *affine combination* of $x_1, ..., x_m$. Moreover, if $\lambda_i \geq 0$ for all $i \in \{1, ..., m\}$, we say that x is a *convex combination* of $x_1, ..., x_m$.

Given a set $S = \{x_1, ..., x_m\} \in \mathbb{R}^{n \times m}$, the *convex hull* of S is the set of point $x \in \mathbb{R}^n$ which are convex combination of $x_1, ..., x_m$ (see Figure 1.2), that is

$$conv(S) = \{x \in \mathbb{R}^n \mid x \text{ is a convex combination of } x_1, ..., x_m\}.$$

The points $x_1, ..., x_m \in \mathbb{R}^n$ are *linearly independant* if the unique solution of the system $\sum_{i=1}^{m} \lambda_i x_i = 0$ is $\lambda_i = 0$, $i = 1, ..., m$. They are *affinely independant* if the unique

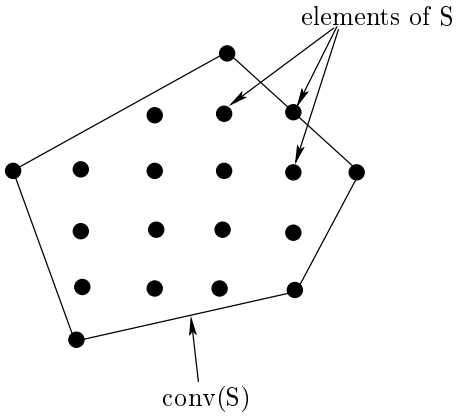


Figure 1.2: A convex hull

solution of the system

$$\begin{cases} \sum_{i=1}^{m} \lambda_i x_i = 0, \\ \sum_{i=1}^{m} \lambda_i = 0, \end{cases}$$

is $\lambda_i = 0$, $i = 1, ..., m$.

A *polyhedron* P is the set of solutions of a linear system $Ax \leq b$, that is $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$, where A is a m-lines n-column matrix and $b \in \mathbb{R}^m$. A *polytope* is a bounded polyhedron. A point x of P will be also called a *solution* of P.

A polyhedron $P \subseteq \mathbb{R}^n$ is said of *dimension* p if the maximum number of solutions of P that are affinely independant is $p + 1$. We denote it by $dim(P) = p$. We also have that $dim(P) = n - rank(A^=)$ where $A^=$ is the submatrix of A of inequalities that are satisfied with equality by all the solutions of P (implicit equalities). The polyhedron P is full dimensional if $dim(P) = n$.

An inequality $ax \leq \alpha$ is *valid* for a polyhedron $P \subseteq \mathbb{R}^n$ if for every solution $\overline{x} \in P$, $a\overline{x} \leq \alpha$. This inequality is said to be *tight* for a solution $\overline{x} \in P$ if $a\overline{x} = \alpha$. The inequality $ax \leq \alpha$ is *violated* by $\overline{x} \in P$ if $a\overline{x} > \alpha$. The set $\mathcal{F} = \{x \in P \mid ax = \alpha\}$ is called a *face* of P. We also say that $\mathcal{F}$ is the *face induced by* $ax \leq \alpha$. If $\mathcal{F} \neq \emptyset$ and $\mathcal{F} \neq P$, we say that $\mathcal{F}$ is a *proper* face of P. If $\mathcal{F}$ is a proper face and $dim(\mathcal{F}) = dim(P) - 1$, then $\mathcal{F}$ is called a *facet* of P. We also say that $ax \leq \alpha$ induces a facet of P or is a *facet defining* inequality.

If P is full dimensional, then $ax \leq \alpha$ is a facet of P if and only if $\mathcal{F}$ is a proper face

and there exists a facet $bx \leq \beta$ of P and a scalar $\rho \neq 0$ such that $\mathcal{F} \subseteq \{x \in P \mid bx = \beta\}$ and $b = \rho a$.

An inequality $ax \leq \alpha$ is *essential* if it defines a facet of P. It is *redundant* if the system $A'x \leq b'$ obtained by removing this inequality from $Ax \leq b$ defines the same polyhedron P. This is the case when $ax \leq \alpha$ can be written as a linear combination of the inequalities of the system $A'x \leq b'$. A *complete minimal linear description* of a polyhedron consists of the system given by its facet defining inequalities and its implicit equalities.

A solution x is an *extreme point* of a polyhedron P if and only if it cannot be written as the convex combination of two different solutions of P. It is equivalent to say that x induces a face of dimension 0. The polyhedron P can also be described by its extreme points. In fact, every solution of P can be written as a convex combination of some extreme points of P. Figure 1.3 illustrates the main definitions given in this section.

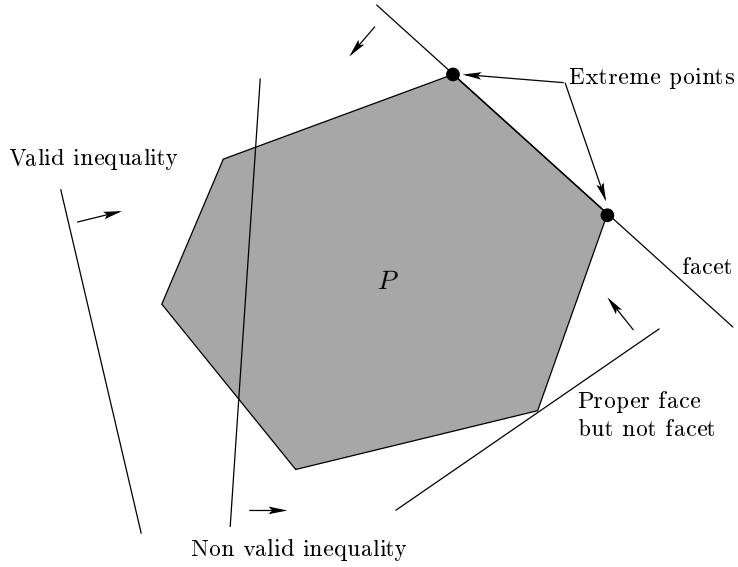


Figure 1.3: Valid inequality, facet and extreme points

1.1.4 Polyhedral approach, Branch-and-Cut method

Here we present the algorithmic aspect of polyhedra and its application to combinatorial optimization problems. Let $\mathcal{P}$ be a combinatorial optimization problem, E its basic set, $w(.)$ the weight function associated with the variables of $\mathcal{P}$ and $\mathcal{S}$ the set of feasible solutions. Suppose that $\mathcal{P}$ consists in finding an element of $\mathcal{S}$ whose

weight is maximum. If $F \subseteq E$, then the 0-1 vector $x^F \in \mathbb{R}^E$ such that $x^F(e) = 1$ if $e \in F$ and $x^F(e) = 0$ otherwise, is called the *incidence vector* of F. The polyhedron $P(\mathcal{S}) = conv\{x^S \mid S \in \mathcal{S}\}$ is the *polyhedron of the solutions of* $\mathcal{P}$ or *polyhedron associated with* $\mathcal{P}$. $\mathcal{P}$ is thus equivalent to the linear program $max\{wx \mid x \in P(\mathcal{S})\}$. The polyhedron $P(\mathcal{S})$ can be described by a set of facet defining inequalities. When all the inequalities of this set are known, then solving P reduces to solve a linear program. The objective of the polyhedral approach for combinatorial optimization problems is to reduce the resolution of $\mathcal{P}$ to that of a linear program. The efficiency of the method thus relies on a deep study of the polyhedron associated with the problem.

However, a complete characterization of the polytope of a problem is difficult to determine. In particular, when the problem is NP-hard there is a little hope to find such a characterization. Moreover, the number of inequalities describing this polyhedron is, in general, exponential. Therefore, even if we know the complete description of that polyhedron, its resolution remains a hard task because of the large number of inequalities.

Fortunately, as it has been shown by Grötschel, Lovász and Schrijver [64], the difficulty for solving a linear program does not depend on the number of inequalities of the program, but on which is called the separation problem associated with the inequality system of the program. Let $Ax \leq b$ be a system of inequalities in $\mathbb{R}^n$. The *separation problem* associated with $Ax \leq b$ is, given $\overline{x} \in \mathbb{R}^n$, to determine whether $\overline{x}$ satisfies $Ax \leq b$ and, if not, to find an inequality $ax \leq \alpha$ of $Ax \leq b$ violated by $\overline{x}$. In 1981, Grötschel, Lovász and Schrijver [64] showed that an optimization problem $\max\{cx,\ Ax \leq b\}$ can be solved in polynomial time if and only if the separation problem associated with $Ax \leq b$ so is. The *cutting plane method* consists in solving a linear program having a large number of inequalities by using the following steps. Let $LP = \max\{cx,\ Ax \leq b\}$ be a linear program and LP' a linear program obtained by considering a small number of inequalities among $Ax \leq b$. Let x^* be its optimal solution. We solve the separation problem associated with $Ax \leq b$ and x^*. This phase is called the *separation phase*. If every inequality of $Ax \leq b$ is satisfied by x^*, then x^* is also optimal for LP. If not, let $ax \leq \alpha$ be an inequality violated by x^*. Then we add $ax \leq \alpha$ it to LP' and repeat this process until an optimal solution is found. Algorithm 1 summarizes the different steps of a cutting plane algorithm.

Algorithm 1: A cutting plane algorithm

Data: A linear program LP and $Ax \leq b$ its system of inequalities
Result: Optimal solution x^* of LP
begin
1 Consider a linear program LP' with a small number of inequalities of LP
2 Solve LP' and let x^* be an optimal solution
3 Solve the separation problem associated with $Ax \leq b$ and x^*
4 **if** *an inequality* $ax \leq \alpha$ *of* LP *is violated by* x^* **then**
5 Add $ax \leq \alpha$ to LP'
6 Repeat step 2
else
7 x^* is optimal for LP
8 return x^*
end

The polyhedron $P(\mathcal{S})$ is often not completely known because $\mathcal{P}$ may be NP-hard. In this case, it would not be possible to solve $\mathcal{P}$ as a linear program. However, one may be able to solve efficiently the linear relaxation of $P(S)$. In general, the solution obtained from the linear relaxation of $P(\mathcal{S})$ is fractional. The resolution of $\mathcal{P}$ can then be done by combining the cutting plane method with a Branch-and-Bound algorithm. Such algorithm is called a *Branch-and-Cut algorithm*. Each node of the Branch-and-Bound tree (also called *Branch-and-Cut tree*) corresponds to a linear program. Suppose that $\mathcal{P}$ is equivalent to $\max\{wx \mid Ax \leq b,\ x \in \{0,1\}^n\}$ and that $Ax \leq b$ has a large number of inequalities. A Branch-and-Cut algorithm starts by creating a Branch-and-Bound tree whose root node corresponds to a linear program $LP_0 = max\{wx \mid A_0x \leq b_0,\ x \in \mathbb{R}^n\}$, where $A_0x \leq b_0$ is a subsystem of $Ax \leq b$ with a small number of inequalities. Then we solve the linear relaxation of $\mathcal{P}$ that is $\overline{LP} = \max\{cx \mid Ax \leq b,\ x \in \mathbb{R}^n\}$, using a cutting plane algorithm starting from the program LP_0. Let x_0^* be its optimal solution and $A_0'x \leq b_0'$ the set of inequalities added to LP_0 at the end of the cutting plane phase. If x_0^* is integral, then it is optimal for $\mathcal{P}$. If x_0^* is fractional, then we start the *branching phase*. This consists in choosing a variable, say x^1, having a fractional value and adding two nodes P_1 and P_2 in the Branch-and-Cut tree. The node P_1 corresponds to the linear program $LP_1 = max\{wx \mid A_0x \leq b_0, A_0'x \leq b_0', x^1 = 0, x \in \mathbb{R}^n\}$ and $LP_2 = max\{wx \mid A_0x \leq b_0, A_0'x \leq b_0', x^1 = 1, x \in \mathbb{R}^n\}$. We solve the linear program $\overline{LP}_1 = \max\{wx \mid Ax \leq b,\ x^1 = 0,\ x \in \mathbb{R}^n\}$ ($\overline{LP}_2 = \max\{wx \mid Ax \leq b,\ x^1 = 1,\ x \in \mathbb{R}^n\}$) by a cutting plane method starting from LP_1 (LP_2). If the optimal solution of $\overline{LP}_1$ ($\overline{LP}_2$) is integral then, it is feasible for $\mathcal{P}$. Its value is thus a lower bound of the optimal solution of $\mathcal{P}$ and the node P_1 becomes a leaf of the Branch-and-Cut tree. If

this solution is fractional, then we select a variable with a fractional value and add two children to the node P_1 (P_2), and so on.

The linear program corresponding to a node of the Branch-and-Cut tree may be infeasible, that is the addition of a constraint $x^i = 0$ or $x^i = 1$ makes the linear program infeasible. Also, even if it is feasible, its optimal solution may be worse than the best known lower bound of the problem. In both cases, we prune that node from the Branch-and-Cut tree. The algorithm ends when all the nodes have been explored. At the end of the algorithm, the optimal solution of $\mathcal{P}$ is the best feasible solution among the solutions given by the Branch-and-Bound tree. Figure 1.4 illustrates the algorithm.

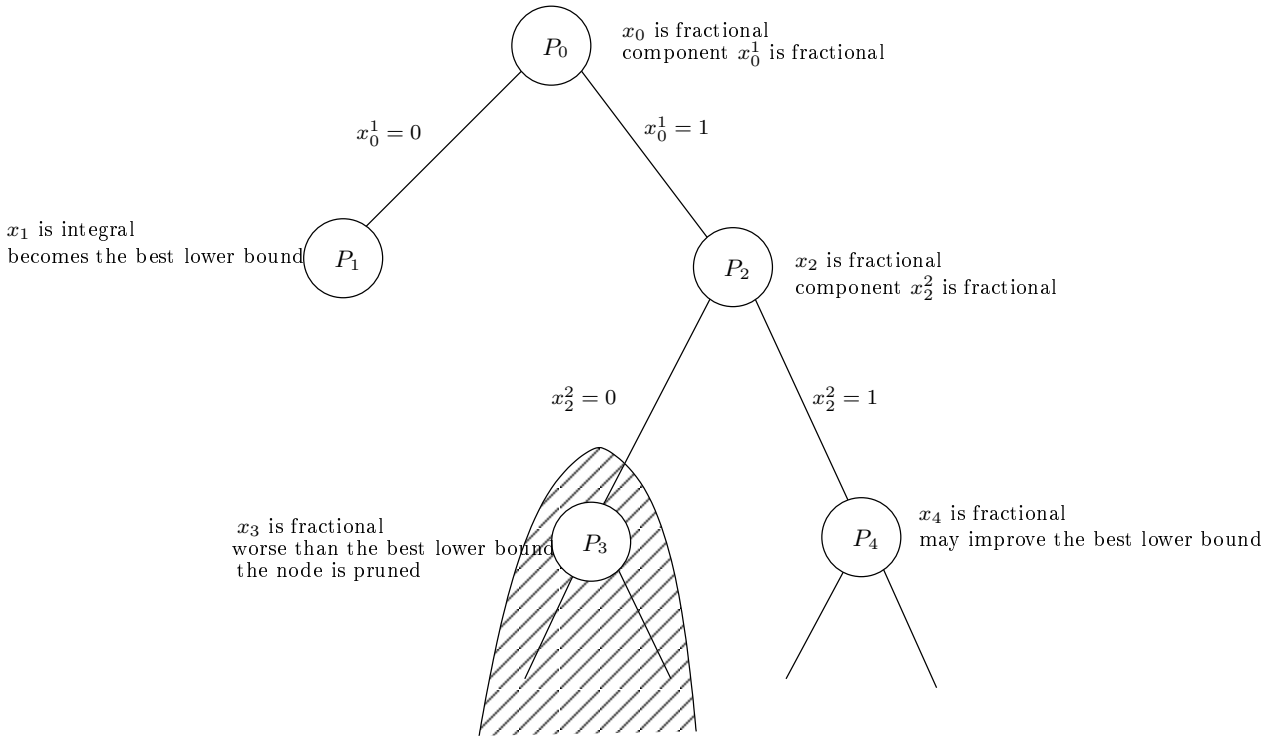


Figure 1.4: A Branch-and-Cut tree.

The algorithm can be improved by computing a good lower bound of the optimal solution of the problem before it starts. This lower bound can be used by the algorithm to prune the nodes which will not allow an improvement of this lower bound. This would permit to reduce the number of nodes generated in the Branch-and-Cut tree and hence reduce the time used by the algorithm. Also, this lower bound can be improved by computing at each node of the Branch-and-Cut tree a feasible solution when the solution obtained at a node is fractional. This is done by using a *primal heuristic*. It

aims to produce a feasible solution for $\mathcal{P}$ from the solution obtained at a given node of the Branch-and-Cut tree, when this later solution is fractional (and hence infeasible for $\mathcal{P}$). Moreover, the weight of this solution must be as best as possible. When the solution computed is better than the best known lower bound, it can considerably reduce the number of generated nodes as well as the CPU time. Moreover, this guarantees to have an approximation of the optimal solution of $\mathcal{P}$ before visiting all the nodes of the Branch-and-Cut tree, for example when a CPU time limit has been reached.

The Branch-and-Cut method is widely used to solve combinatorial optimization problems that are considered difficult to solve, such as the Travelling Salesman Problem [4]. Its efficiency can be considerably increased by a good knowledge of the polyhedron associated with the problem and by efficient separation algorithms. The cutting plane method is effective when the number of variables is polynomial. However, when the number of variables is large (for example exponential), other methods, such as the column generation method, are more appropriate to use. In the following section we briefly describe this method.

1.1.5 Column generation and Branch-and-Cut-and-Price methods

The *column generation method* is used to solve linear programs with a large number of variables. The method aims to solve the linear program by considering a small number of variables. This method was introduced by Dantzig and Wolfe [36] in 1960 in order to solve linear programs with large number of variables by using few ressources (CPU time and memory consumption). The column generation method is used either for problems which can be solved using *Dantzig-Wolfe decomposition method* or for problems with a large number of variables.

The idea of a column generation algorithm is to solve a sequence of linear programs having a reasonable number of variables (also called *columns*). The algorithm starts by solving a linear program having a small number of variables and which forms a feasible basis for the original program. At each iteration of the algorithm, we solve the so-called *pricing problem* whose objective is to determine the variables which must enter the current basis. These variables are those having a negative reduced cost. The *reduced cost* associated with a variable is computed using the dual variables. We then solve the linear program obtained by the addition of these variables and repeat the procedure. The algorithm stops when the pricing algorithm does not generate new column to add in the current basis. In this case, the solution obtained from the last restricted program is optimal for the original one.

The column generation method can be seen as the dual of the cutting plane method as it adds columns (variables) instead of rows (inequalities) in the linear program. The pricing problem can be NP-hard. In this case, one can use heuristic procedures to solve it. For more details on column generation algorithms, the reader is referred to [85, 86, 102].

In order to solve integer linear programs, the column generation method can be combined with a Branch-and-Bound algorithm. In this case, the algorithm is called a *Branch-and-Price algorithm*. The branching phase happens when no variable can be added into the current linear program and the solution given by that program is fractional. Moreover, the algorithm can be combined with a cutting plane algorithm, that looks for inequalities that are valid for the problem but violated by the current fractional solution. These can be added to the current linear program. In this case, we speak of *Branch-and-Cut-and-Price algorithm*. Barnhart et al. [9] use this technique to solve large scale integer multicommodity flow problems. Barhnart et al. [10] present huge problems which have been solved using Branch-and-Price method.

1.1.6 Graph theory: notations and definitions

In this section, we present some basic definitions and notations of graph theory which will be frequently used in the subsequent chapters. For more details, the reader is referred to [15].

The graphs we consider are either directed or undirected, finite, loopless and may contain multiple arcs or edges.

An undirected graph is denoted by $G = (V, E)$ where V is the set of nodes and E is the set of edges. If $e \in E$ is an edge with endnodes u and v, we also write uv to denote e. For a node subset $W \subseteq V$, we denote by $\overline{W}$ the node set $V \setminus W$. Given W and W', two disjoint subsets of V, $[W, W']$ denotes the set of edges of G having one endnode in W and the other one in W'. If $W' = \overline{W}$, then $[W, \overline{W}]$ is called a *cut* of G and denoted by $\delta(W)$. A cut $\delta(W)$ is said to be *proper* if $|W| \geq 2$ and $|\overline{W}| \geq 2$. If $\pi = (V_1, ..., V_p)$, $p \geq 2$, is a partition of V, then we denote by $\delta(\pi)$ the set of edges having their endnodes in different sets. We may also write $\delta(V_1, ..., V_p)$ for $\delta(\pi)$. Note that for $W \subset V$, $\delta(W) = \delta(W, \overline{W})$.

A directed graph is denoted by $H = (U, A)$ where U is the node set and A the arc set. An arc a with origin u and destination v is denoted by (u, v). Given two node subsets W and W' of U, $[W, W']$ denotes the set of arcs whose origins are in W and

destinations are in W'. As before, we write $[u, W']$ for $[\{u\}, W']$ and $\overline{W}$ denotes the node set $U \setminus W$. The set of arcs having their origin in W and destination in $\overline{W}$ is called a *directed cut* or *dicut* of H. This arc set is denoted either by $\delta^+(W)$ or $\delta^-(\overline{W})$. We also write $\delta^+(u)$ for $\delta^+(\{u\})$ and $\delta^-(u)$ for $\delta^-(\{u\})$ with $u \in U$. If s and t are two nodes of H such that $s \in W$ and $t \in \overline{W}$, then $\delta^+(W)$ and $\delta^-(\overline{W})$ are called an *st-dicuts* of H.

Let $G' = (V', E')$ (resp. $H' = (U', A')$) with $V' \subseteq V$ and $E' \subseteq E$ (resp. $U' \subseteq U$ and $A' \subseteq A$) be a subgraph of G (resp. H). If $w(.)$ is a weight function which associates with each edge (resp. arc) $e \in E$ (resp. $a \in A$) the weight $w(e)$ (resp. $w(a)$), then the total weight of G' (resp. H') is $w(E') = \sum_{e \in E'} w(e)$ (resp. $w(A') = \sum_{e \in A'} w(a)$).

In the notation, we will specify the graph as a subscript (that is, we will write $\delta_G(W)$, $\delta_G(\pi)$, $\delta_G(V_1, ..., V_p)$, $\delta^+_H(W)$, $\delta^-_H(W)$, $[W, W']_G$, $[W, W']_H$) whenever the considered graphs may not be clearly deduced from the context.

Given an undirected graph $G = (V, E)$, for all $F \subseteq E$, $V(F)$ will denote the set of nodes incident to the edges of F. For $W \subset V$, we denote by $E(W)$ the set of edges of G having both endnodes in W and $G[W]$ the subgraph induced by W, that is the graph $(W, E(W))$. Given an edge $e = uv \in E$, *contracting* e consists in deleting e, identifying the nodes u and v and in preserving all adjacencies. Contracting a node subset W consists in identifying all the nodes of W and preserving the adjacencies. Given a partition $\pi = (V_1, ..., V_p)$, $p \geq 2$, we will denote by G_π the subgraph induced by π, that is, the graph obtained from G by contracting the sets V_i, for $i = 1, ..., p$. Note that the edge set of G_π is the set $\delta(V_1, ..., V_p)$.

A *Path* P of an undirected graph G is an alternate sequence of nodes and edges $(u_1, e_1, u_2, e_2, ..., u_{q-1}, e_{q-1}, u_q)$ where $e_i \in [u_i, u_{i+1}]$ for $i = 1, ..., q-1$. We will denote a path P either by its node sequence $(u_1, ..., u_q)$ or its edge sequence $(e_1, ..., e_{q-1})$. The nodes u_1 and u_q are called the *endnodes* of P, while its other nodes are said to be *internal*. A path is *simple* if it does not contain the same node twice. In the sequel, we will always consider that the paths are simple. By opposition, a non-simple path is called a *walk*. A path whose endnodes are s and t will be called an *st-path*. A *cycle* in G is a path whose endnodes coincide, that is $u_1 = u_q$. Also, a cycle is simple if it does not contain twice the same node, excepted u_1. We call a *chord* an edge between two non-adjacent nodes of a path.
Similarly, we call a *dipath* P a path in a directed graph, that is an alternate sequence of arcs $(u_1, a_1, u_2, a_2, ..., u_{q-1}, a_{q-1}, u_q)$ with $a_i \in [u_i, u_{i+1}]$, $i = 1, ..., q-1$. A dipath is denoted either by its node sequence $(u_1, ..., u_q)$ or its arc sequence $(a_1, ..., a_{q-1})$, and the nodes u_1, u_q are the endnodes of that dipath. A dipath whose endnodes coincide

$(u_1 = u_q)$ is called a *circuit*. If $u_1 = s$ and $u_q = t$ then P is called an *st-dipath*. A dipath is simple if it does not contain twice the same node.
Given a fixed integer $L \geq 1$ and a pair of nodes $\{s,t\} \in V \times V$, an *L-st-path* in G is a path between s and t whose length is at most L, where the *length* is the number of edges of that path. The number of edges of a path is also called *hops* and we also speak of *L-hop-constrained* paths for paths whose length is at most L.

An undirected (resp. directed) graph is *connected* if for every pair of node (u,v) there is at least one path (resp. dipath) between u and v. A connected graph which have no cycle (resp. circuit) is called a *spanning tree*. A *connected component* of a graph G (resp. H) is a connected subgraph of G (resp. H) which is maximal, that is adding a node or an edge (resp. arc) to that subgraph gives a non-connected graph.

Given an undirected (resp. directed) graph $G = (V,E)$ (resp. $H = (U,A)$), two *st*-paths (resp. *st*-dipaths) are *edge-disjoint* (resp. *arc-disjoint*) if they have no edge (resp. arc) in common. They are *node-disjoint* if they have no internal node in common. A graph is said to be *k-edge-connected* (resp. *k-arc-connected*) if it contains at least k edge-disjoint (resp. arc-disjoint) *st*-paths (resp. *st*-dipaths) for all pair of node $\{s,t\} \in V \times V$ (resp. $\{s,t\} \in U \times U$). It is *k-node-connected* if it contains at least k node-disjoint *st*-paths or *st*-dipaths for all pair of node $\{s,t\} \in V \times V$ (resp. $\{s,t\} \in U \times U$). The largest integer k such that the graph G (resp. H) is k-edge-connected (resp. k-arc-connected) is the *edge-connectivity* (resp. *arc-connectivity*) of G (resp. H). Similarly, the largest integer k such that the graph is k-node-connected is the *node-connectivity* of the graph. We say that a graph is *Steiner* k-edge-connected (k-arc-connected) (k-node-connected) if it is k-edge-connected (k-arc-connected) (k-node-connected) relatively to a certain pair of privileged nodes. We ommit the qualificative Steiner when the required connectivity is for every pair of nodes of the graph. The privileged nodes are called *terminal nodes* while non-privileged ones are called *Steiner nodes*.

Given an undirected graph $G = (V,E)$, a *demand set* $D \subseteq V \times V$ is a subset of pairs of nodes, called *demands*. For a demand $\{s,t\} \in D$, s is the *source* of the demand and t is the *destination* of that demand. If several demands $\{s,t_1\}, ..., \{s,t_d\}$ have the same node s as source node, then these demands are *rooted* in s. A node involved in at least one demand is said to be *terminal*. A node which does not belong to any demand is called a *Steiner node*.

A *complete graph* is a graph in which there is an edge between each node and the others. A complete graph with n nodes is denoted by K_n. A *bipartite* graph $G = (V,E)$ is an undirected graph such that $V = V_1 \cup V_2$ with $V_1 \cap V_2 = \emptyset$ and for every pair of nodes $u,v \in V_1$ (resp. $u,v \in V_2$), $[u,v] = \emptyset$. A *complete bipartite graph* is a bipartite

graph where there is an edge between each node of V_1 and the nodes of V_2. A bipartite complete graph is denoted $K_{m,n}$ where $m = |V_1|$ and $n = |V_2|$.
An undirected graph is *outerplanar* when it can be drawn in the plane as a cycle with non crossing chords. A graph is *series-parallel* if it can be obtained from a single edge by iterative application of the two operations:

i) addition of a parallel edge;

ii) subdivision of an edge.

Observe that a graph is series-parallel (outerplanar) if and only if it is not contractible to K_4 (K_4 and $K_{3,2}$). Therefore, an outerplanar graph is also series-parallel.
A graph G is said to be a *Halin graph* if $G = (C \cup T, E)$ where the subgraph of G induced by T is a tree whose leaves forms the cycle C in G. Figure 1.5 gives an example of each type of graphs described above.

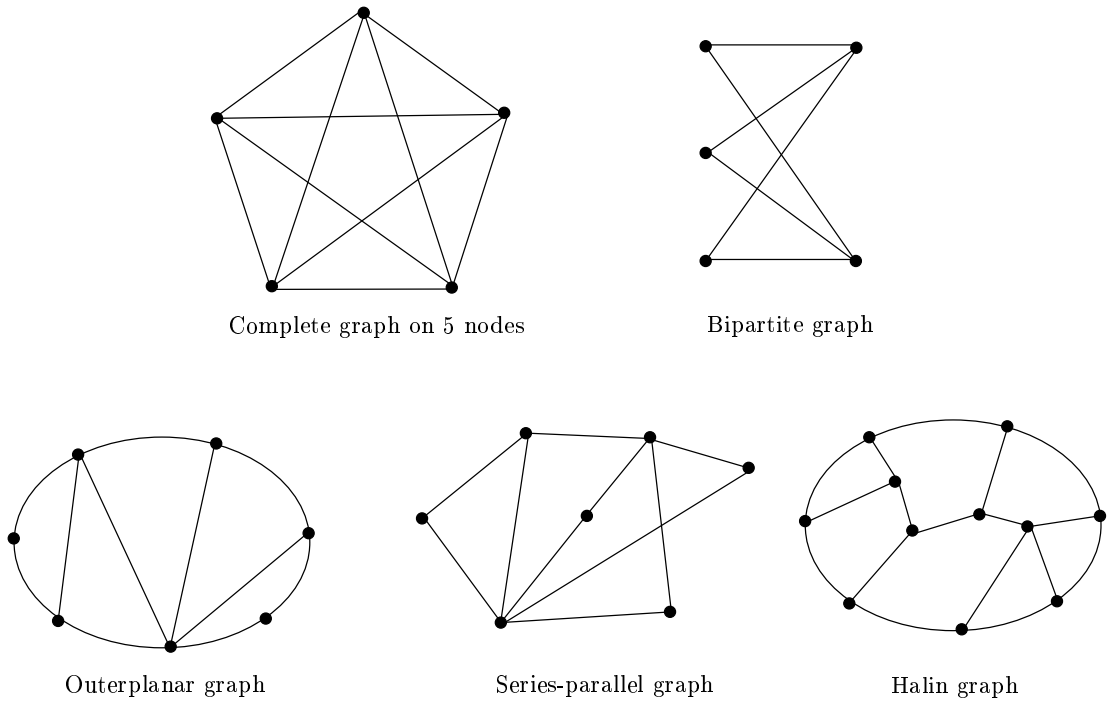


Figure 1.5: Complete, bipartite, outerplanar, series-parallel and Halin graphs.

1.2 State-of-the-art on survivable network design problems

Survivable network design problems have been intensively studied for several decades. The first studies on the problems aimed to produce heuristics and approximation algorithms for these problems. Since the begining of 90's, studies starts focusing on exact algorithms with, in particular, the use of the polyhedral approach.

This section is dedicated to the presentation of the previous works in the litterature related to survivable network design problems. We first present the general survivable network design problem, the related works and main results on this problem. Then we discuss two variants of the problem, the k-edge-connected subgraph problem and the k-edge-connected hop-constrained network design problem. These will be studied in Chapters 2 and 3 for the first one and Chapters 4, 5 and 6 for the second one.

1.2.1 The general survivable network design problem

A network can be represented by a graph, directed or undirected, where each node of the network corresponds to a node of the graph and a link between two nodes of the network is represented by an edge or an arc of the graph.

Consider an undirected graph $G = (V, E)$ representing a telecommunication network and $w(.)$ a weight function which associates the weight $w(e)$ with an edge $e \in E$. Each node $v \in V$ is associated with an integer, denoted by $r(v)$ and called *connectivity type* of v, which can be seen as the minimum number of edges connecting v to the rest of the network. The vector $(r(v) \mid v \in V)$ is the connectivity type vector associated with the nodes of G. We say that a subgraph $H = (U, F)$, $U \subseteq V$ and $F \subseteq E$, satisfies the *edge-connectivity* (resp. *node-connectivity*) *requirement* if for every pair of nodes $(s, t) \in V \times V$, there exist at least

$$r(s, t) = min\{r(s), r(t)\}$$

edge-disjoint (resp. node-disjoint) paths between s and t. This condition ensures that the network will be still functional when certain equipment fails. In fact, the traffic can still be routed between two nodes s and t when at most $r(s, t) - 1$ links, in case of edge-connectivity, and at most $r(s, t) - 1$ nodes, in case of node-connectivity, fails. When $r(u) = k$, for every $u \in V$, the subgraph H is k-edge-connected (resp. k-node-connected).

Let $r_{max} = max\{r(u) \mid u \in V\}$. When $r_{max} \leq 2$ we speak of *low connectivity requirement* and of *high connectivity requirement* when $r_{max} \geq 3$.

Grotschel, Monma and Stoer [66] introduced the *general survivable network design problem* which consists in finding a minimum weight subgraph of G which satisfies the connectivity requirement. We will denote this problem by *ESNDP* (resp. *NSNDP*) for edge-connectivity (resp. node-connectivity) requirement.

The ESNDP (NSNDP) is NP-hard as it contains the Steiner tree problem as a special case ($r(u) \in \{0,1\}$ for all $u \in V$) which is known to be NP-hard [56]. However, under certain conditions the problem can be solved in polynomial time. When $r(u) = 1$ for all $u \in V$, the problem is equivalent to the minimum weight spanning tree problem. Thus it is solvable in polynomial time using Kruskal [84] or Prim [95] algorithms. Also when $r(s) = r(t) = 1$ for two nodes $s, t \in V$ and $r(u) = 0$ for all $u \in V \setminus \{s,t\}$, the problem is nothing but the shortest st-path problem which can be solved in polynomial time with the effecient algorithm of Dijkstra [43].

Menger [91] exhibited the relation between the number of edge-disjoint paths and the cardinality of cuts in the graph G. This relation is given in the theorem below.

Theorem 1.2.1 [91, 96] *Let $G = (V, E)$ be an undirected graph and s, t two nodes of G. Then, there exist at least k edge-disjoint paths between s and t if and only if every st-cut of G contains at least k edges.*

By Theorem 1.2.1, the ESNDP can be described as a linear integer program. To this end let us introduce first some notations.

$$\begin{array}{lcll} r(W) & = & max\{r(u) \mid u \in W\} & \text{for all } W \subseteq V, \\ con(W) & = & max\{r(u,v) \mid u \in W, v \in \overline{W}\} & \\ & = & min\{r(W), r(\overline{W})\} & \text{for all } W \subseteq V, \emptyset \neq W \neq V. \end{array}$$

The ESNDP is equivalent to the following linear integer program.

$$\begin{array}{lll} \text{Minimize} \sum_{e \in E} c(e)x(e) & & \\ x(\delta(W)) \geq con(W) & \text{for all } W \in V, \emptyset \neq W \neq V, & (1.1) \\ x(e) \geq 0 & \text{for all } e \in E, & (1.2) \\ x(e) \leq 1, & \text{for all } e \in E & (1.3) \\ x(e) \in \{0,1\} & & (1.4) \end{array}$$

Grötschel and Monma [65] study the polyhedral aspects of that model. They discuss the dimension of the associated polytope as well as some basic facets. In [66], Grötschel et al. study further polyhedral aspects of that model. They devise cutting plane algorithms and give computational results.

In [57], Goemans and Bertsimas give an approximation algorithm based for the ES-NDP based on a new analysis of a well-known algorithm for the Steiner tree problem.

A related problem is the so-called augmentation problem. Given an undirected graph $G = (V, E)$ and a connectivity vector $(r(v) \mid v \in V)$, the *augmentation problem* is to add as few edges as possible to G so that the resulting graph satisfies the connectivity requirements given by r. This problem is equivalent to the general survivable network design problem on a complete graph where the weight of the edges of E is 0 and that of the edges that can be added is 1. Eswaran and Tarjan [47] studied that problem in the cases where $r(u) = 2$ for all $u \in V$. They gave polynomial time algorithms for the cases where edge-disjoint and node-disjoint paths are required. Watanabe and Nakamura [103] and Cai and Sun [18] studied the problem when $r(u) = k$ for all $u \in V$ and $k \in 2$. They [18, 103] gave polynomial time algorithms for the problem in that case. Cai and Sun [18] also gave a min-max formula for the minimum number of edges that must be added. Frank [53] considered the problem for an arbitrary connectivity vector $r \in \mathbb{N}^V$. Using the splitting theorem of Mader [87], he gave a min-max formula for the minimum number of edges that must be added to the original graph and devise a polynomial time algorithm for the problem. Its results generalize those obtained by [47] and [18].

1.2.2 The k-edge(node)-connected subgraph problem

The k-edge-connected subgraph problem has been extensively studied, especially when $k = 2$ (low connectivity requirement) [8, 49, 54, 80, 81, 83, 88, 89, 92]. However, it has received a little attention in the case where $k \geq 3$.

In [21], Chopra studied the problem for k odd when multiple copies of an edge may be used. In particular, he characterized the associated polyhedron for outerplanar graphs. This polyhedron has been previously studied by Cornuéjols et al. [23]. They characterized the associated polytope when the graph is series-parallel and $k = 2$. In [40], Didi Biha and Mahjoub also studied the problem when the graph is series-parallel and $k \geq 3$, and gave a complete description of the polytope in that case. In [49], Fonlupt and Mahjoub studied the fractional extreme points of the linear relaxation of the 2-edge-connected subgraph polytope. They introduced an ordering on these

extreme points and characterized the minimal extreme points with respect to that ordering. As a consequence, they obtained a characterization of the graphs for which the linear relaxation of that problem is integral. Didi Biha and Mahjoub [39], extended some of the results of Fonlupt and Mahjoub [49] to the case $k \geq 3$ and introduced some graph reduction operations.

Much work has been done on the problem when $k = 2$. In [7], Baïou and Mahjoub study the Steiner 2-edge-connected subgraph polytope. This has been generalized by Didi Biha and Mahjoub [41] to the Steiner k-edge-connected subgraph polytope for k even. Mahjoub [88] introduces a general class of valid inequalities for the polytope of the problem when $k = 2$. Boyd and Hao [17] describe a class of "comb inequalities" which are valid for 2-edge-connected subgraph polytope. This class, as well as that introduced by Mahjoub [88], are special cases of a more general class of inequalities given by Grötschel et al. [66] for the general survivable network polytope. In [8], Barahona and Mahjoub characterize the 2-edge-connected subgraph polytope for the class of Halin graphs. Kerivin et al. [81] describe a general class of valid inequalities for the problem that generalizes the so-called F-partition inequalities introduced by [88]. They also develop a Branch-and-Cut algorithm for the problem. In [25, 26], Coullard et al. study the Steiner 2-node-connected subgraph problem. They devise in [25] a linear time algorithm for this problem on some special classes of graphs. Moreover in [26], they characterize the dominant of the polytope associated with this problem on the graphs which do not have K_4 as a minor.

Monma et al. [92] described some structural properties of the optimal solution of the k-edge-connected subgraph problem when the cost function satisfies the triangle inequalities (*i.e.*, $c(e_1) \leq c(e_2)+c(e_3)$ for every three edges e_1, e_2, e_3 defining a triangle). In particular, they showed that every node of a minimum weight k-edge-connected subgraph has degree 2 or 3. They also showed that the cost of an optimal tour solution of the TSP (Travelling Salesman Problem) is at most $\frac{4}{3}$ times the cost of an optimal solution of the 2-edge-connected subgraph problem. They [92] devised a heuristic based on these properties. Bienstock et al. [14] extended the result obtained by [92] to the case where $k \geq 3$ and showed that every node of a minimum cost k-edge-connected subgraph has degree k or $k + 1$. This result also generalizes the result obtained by Frederickson and Jájá [54]. In [82], Khuller and Raghavachari gave an approximation algorithm for the smallest k-edge-connected subgraph problem ($c(e) = 1$ for all $e \in E$). They proved that the cost of a solution given by their algorithm is at most 1.85 of the optimal solution for all $k \geq 2$. Fernandes [48] showed that the ratio of the algorithm of [82] is, in fact, 1.75 for all $k \geq 2$. The algorithm is the first algorithm to achieve a performance ratio less than 2. They [82] also gave an approximation algorithm for the minimum cost k-node-connected subgraph problem with $k \geq 2$ in the case where the

cost function satisfies the triangle inequalities. The performance ratio of their algorithm is $2+\frac{2(k-1)}{n}$ where n is the number of nodes of the graph. In [19], Cheriyan et al. gave an $\frac{17}{12}$-approximation algorithm for the 2-edge-connected subgraph problem. Cheriyan and Thurimella [20] gave a $(1+\frac{2}{k+1})$-approximation algorithm for the smallest k-edge-connected subgraph problem with $k \geq 2$. Karger [78] gave a randomized algorithm for the smallest k-edge-connected subgraph problem. He proved that the performance ratio of its algorithm is $1+O(\sqrt{\frac{\log n}{k}})$. Gabow et al. [55] introduced a approximation algorithm for the k-edge-connected subgraph problem based on LP-rounding. They showed that for undirected graphs the ratio of the LP-rounding algorithm is $1+\frac{3}{k}$ when k is odd and $1+\frac{2}{k}$ when k is even.

The directed version of the Steiner k-edge-connected subgraph problem has also been studied. This problem is described as follows. Let $H=(U,A)$ be a directed graph, $D \subseteq U \times U$ be a set of demands and a weight function $w(.)$ which associates the weight $w(a)$ with each arc of H. Given an integer $k \geq 2$, the *Survivable Directed Network Design Problem* (kDNDP for short) consists in finding a minimum cost subgraph of H which contains k-arc-disjoint st-dipaths for all $\{s,t\} \in D$. This problem has been studied by Suurballe [100] and Soenoka et al. [98]. Suurballe [100] considered the kDNDP when $|D|=1$. The problem can be formulated in this case as a network flow problem, and hence, can be solved using for example network simplex. Suurballe [100] gave a polynomial combinatorial optimization algorithm for the problem in this case. In [98], Soenoka et al. considered the problem of finding a directed k-arc-connected graph with a minimal number of arcs and small diameter (the diamater is the largest among all shortest path lengths, when all the arcs have length 1). Dahl [27, 28, 29] also studied the problem from a polyhedral point of view. In [29], he described several valid inequalities for the polytope of the problem and devised a cutting plane algorithm.

1.2.3 The k-edge-connected hop-constrained network design problem

Given an undirected graph $G=(V,E)$, a weight function $w(.)$, a set of demands $D \subseteq V \times V$ and two integers k, L greater than 2, the *k-edge-connected hop-constrained network design problem* consists in finding a subgraph of G of minimum weight such that for every pair $\{s,t\} \in D$, there exist at least k edge-disjoint paths of length at most L between s and t.

This problem takes some importance since the connectivity requierement is often insufficient regarding the reliability of a telecommunications network. In fact, the

alternative paths could be too long to guarantee an effective routing. In data networks, such as Internet, the elongation of the route of the information could cause a strong loss in the transfer speed. For other networks, the signal itself could be degraded by a longer routing. In such cases, the L-path requirement guarantees exactly the needed quality of the alternative routes.

The k-edge-connected hop-constrained network design problem is a generalization of the k-edge-connected subgraph problem. In fact, this later problem corresponds to the first one in the case where $L = |V| - 1$ and $D = V \times V$.

The k-edge-connected hop-constrained network design problem has been studied in some special cases. Huygens et al. [75] have investigated the case where $k = 2$, $|D| = 1$ and the bound L on the length of the paths is 2 or 3. They give an integer programming formulation for the problem and show that the linear relaxation of this formulation completely describes the polytope associated to the problem in this case. From this, they obtain a minimal linear description of that polytope. They also show that this formulation is no longer valid when $L \geq 4$. In [35], Dahl et al. study the problem when $L = 2$, $k \geq 2$ and $|D| = 1$. They give a complete description of the associated polytope. There has been however a considerable amount of research on many related problems.

In [31], Dahl considers the k-edge-connected hop-constrained path problem, that is the problem of finding between two distinguished nodes s and t a minimum cost path with no more than L edges when L is fixed. He gives a complete description of the dominant of the associated polytope when $L \leq 3$. Thus this hop-constrained path problem corresponds to the special case $k = 1$ and $|D| = 1$ of the k-edge-connected hop-contrained network design problem. Dahl and Gouveia [32] consider the directed hop-constrained path problem. They describe valid inequalities and characterize the associated polytope when $L \leq 3$. Huygens and Mahjoub [73] study the problem when $L \geq 4$ and $|D| = 1$. They also study the variant of the problem where k node-disjoint paths of length at most L are requiered between two terminals s and t. They give an integer programming formulation for these two problems in the case $k = 2$ and $L = 4$.

The case where several pairs (s, t) of terminals have to be linked by L-hop-constrained paths has also been studied in the litterature. In [34], Dahl and Johannessen consider the 2-path network design problem which consists in finding a minimum cost subgraph connecting each pair of terminal nodes by at least one path of length at most 2. The problem of finding a minimum cost spanning tree with hop-constraints is also considered in [60], [61] and [63]. Here, the hop-constraints limit to a positive integer H the number of links between the root and any terminal in the network. Dahl [30] studies

the problem where $H = 2$ from a polyhedral point of view and gives a complete description of the associated polytope when the graph is a wheel. Finally, Huygens et al. [76] consider the problem of finding a minimum cost subgraph with at least two edge-disjoint L-hop-constrained paths between each given pair of terminal nodes. They give an integer programming formulation of that problem for $L = 2, 3$ and present several classes of valid inequalities. They also devise a Branch-and-Cut algorithm, and discuss some computational results. In [24], Coullard et al. investigate the structure of the polyhedron associated with the st-walks of length K of a graph, where a walk is a path that may go through the same node more than once. They present an extended formulation of the problem, and, using projection, they give a linear description of the associated polyhedron. They also discuss classes facets of that polyhedron.

Besides hop-constraints, another reliability condition, which is used in order to limit the length of the routing, requires that each link of the network belongs to a ring (cycle) of bounded length. In [52], Fortz et al. consider the 2-node connected subgraph problem with bounded rings. This problem consists in finding a minimum cost 2-node connected subgraph (V, F) such that each edge of F belongs to a cycle of length at most L. They describe several classes of facet defining inequalities for the associated polytope and devise a Branch-and-Cut algorithm for the problem. In [51], Fortz et al. study the edge version of that problem. They give an integer programming formulation for the problem in the space of the natural design variables and describe different classes of valid inequalities. They study the separation problem of these inequalities and discuss Branch-and-Cut algorithm.

Chapter 2

The k-Edge-Connected Subgraph Problem

In this chapter we consider the k-edge-connected subgraph problem from a polyhedral point of view. We first present an integer programming formulation for the problem. We then introduce further classes of valid inequalities for the associated polytope, and describe sufficient conditions for these inequalities to be facet defining. In Chapter 3 we discuss the algorithmic aspect of this study. We devise separation heuristics for the valid inequalities and discuss some reduction operations that can be used in a preprocessing phase for the separation. Then we develop a Branch-and-Cut algorithm using these results and present some computational results. This work has led to an article that will be published in Networks [12].

2.1 Introduction

Given an undirected graph $G = (V, E)$, an integer $k \geq 2$ and a weight function $w(.)$ which associates with each edge $e \in E$ the weight $w(e) \in \mathbb{R}$, the *k-edge-connected subgraph problem* (kECSP for short) is to find a subgraph $H = (V, F)$ of G such that $\sum_{e \in F} w(e)$ is minimum.

Remind that, given an edge subset $F \subseteq E$, the 0-1 vector $x^F \in \mathbb{R}^E$ such that $x^F(e) = 1$ if $e \in F$ and 0 if $e \in E \setminus F$ is called the incidence vector of F. Let $k\text{ECSP}(G)$ be the convex hull of the incidence vectors of the k-edge-connected subgraphs of G,

that is

$k\text{ECSP}(G) = \text{conv}\{x^F \in \mathbb{R}^E \mid F \subseteq E \text{ and } (V, F) \text{ is a } k\text{-edge-connected subgraph of } G\}$.

If x^F is the incidence vector of the edge set F of a k-edge-connected subgraph of G, then x^F satisfies the following inequalities:

$$x(e) \geq 0 \qquad \text{for all } e \in E, \tag{2.1}$$
$$x(e) \leq 1 \qquad \text{for all } e \in E, \tag{2.2}$$
$$x(\delta(W)) \geq k \quad \text{for all } W \subset V, W \neq V, W \neq \emptyset. \tag{2.3}$$

Conversely, any integer solution of the system defined by inequalities (2.1)-(2.3) is the incidence vector of the edge set of a k-edge-connected subgraph of G. Constraints (2.1) and (2.2) are called *trivial inequalities* and constraints (2.3) are called *cut inequalities*. We will denote by $P(G, k)$ the polytope given by inequalities (2.1)-(2.3).

2.2 Facets of kECSP(G)

In this section we present three classes of valid inequalities for $k\text{ECSP}(G)$. We describe some conditions for these inequalities to be facet defining. But first, we give the following lemmas, which will be frequently used in this section.

Lemma 2.2.1 *If an inequality $ax \geq \alpha$ is different from the trivial inequalities and defines a facet of kECSP(G), then $a(e) \geq 0$ for all $e \in E$ and $\alpha > 0$.*

Proof. Suppose that $a(e) < 0$ for some edge $e \in E$. As $ax \geq \alpha$ is different from the trivial inequality $x(e) \leq 1$, there must exist a solution $F \subseteq E$ of the kECSP which does not contain e and such that $ax^F = \alpha$. Let $F' = F \cup \{e\}$. Obviously, F' also induces a solution of the kECSP. However, since $a(e) < 0$, we have that $ax^{F'} = ax^F + a(e) < \alpha$, contradiction.

In consequence, $a(e) \geq 0$ for all $e \in E$. Moreover, since $ax \geq \alpha$ is facet defining, one should have $a(f) > 0$ for at least one edge f of E. As $ax \geq \alpha$ is different from $x(f) \geq 0$, there exists a solution $\tilde{F}$ of the kECSP which contains f and such that $ax^{\tilde{F}} = \alpha$. This yields $\alpha > 0$. □

Lemma 2.2.2 *Let $G = (V, E)$ be a k-edge-connected graph and $e_0 = u_0v_0$ be an edge of G such that every cut $\delta(U)$ of G containing e_0, except eventually $\delta(u_0)$, is such that $|\delta(U)| \geq k + 1$. If G' is a graph obtained from G by deleting e_0 and adding an edge f incident to u_0, then G' is k-edge-connected.*

Proof. Let $\delta_{G'}(U')$ be a cut of G'. If $\delta_{G'}(U')$ does not separate u_0 and v_0, then, as G is k-edge-connected, we have that $|\delta_{G'}(U')| \geq k$. If this is not the case and $U' \neq \{u_0\}$, then $\delta_G(U')$ contains at least $k + 1$ edges and hence $|\delta_{G'}(U')| \geq k$. Finally, if $U' = \{u_0\}$, since G is k-edge-connected and $\delta_{G'}(u_0) = (\delta_G(u_0) \setminus \{e_0\}) \cup \{f\}$, we have that $|\delta_{G'}(u_0)| \geq k$. □

2.2.1 Odd path inequalities

Let $G = (V, E)$ be a $(k + 1)$-edge connected graph and $\pi = (W_1, W_2, V_1, ..., V_{2p})$ a partition of V with $p \geq 2$. Let $I_1 = \{4r, 4r + 1, r = 1, ..., \lceil \frac{p}{2} \rceil - 1\}$ and $I_2 = \{2, ..., 2p - 1\} \setminus I_1$. We say that π induces an *odd path configuration* if

1. $|[V_i, W_j]| = k - 1$ for $(i, j) \in (I_1 \times \{1\}) \cup (I_2 \times \{2\})$,
2. $|[W_1, W_2]| \leq k - 1$,
3. $\delta(V_i) = [V_i, W_1] \cup [V_{i-1}, V_i] \cup [V_i, V_{i+1}]$ (resp. $\delta(V_i) = [V_i, W_2] \cup [V_{i-1}, V_i] \cup [V_i, V_{i+1}]$) if $i \in I_1$ (resp. $i \in I_2$),
4. $\delta(V_1) = [W_1, V_1] \cup [V_1, V_2]$ and $\delta(V_{2p}) = [W_1, V_{2p}] \cup [V_{2p-1}, V_{2p}]$ (resp. $\delta(V_{2p}) = [W_2, V_{2p}] \cup [V_{2p-1}, V_{2p}]$) if p is even (resp. odd) (see Figure 2.1 for $k = 3$ and p even).

Note that by conditions 3) and 4), we have that $[V_l, V_t] = \emptyset$ for all $l, t \in \{1, ..., 2p\}$ and $|l - t| > 1$.
Let $C = \bigcup_{i=1}^{2p-1} [V_i, V_{i+1}]$. Thus C can be seen as an odd path of extremities V_1 and V_{2p} in the graph G_π. With an odd path configuration we associate the inequality

$$x(C) \geq p. \tag{2.4}$$

Inequalities of type (2.4) will be called *odd path inequalities*. We have the following.

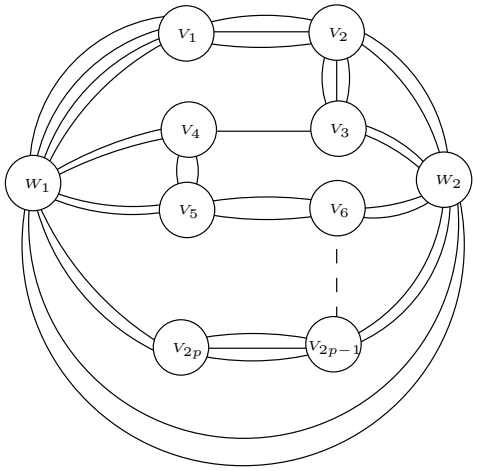


Figure 2.1: An odd path configuration with $k = 3$ and p even.

Theorem 2.2.1 *Inequality (2.4) is valid for kECSP(G).*

Proof. As $|[V_i, W_j]| = k - 1$ and $x(\delta(V_i)) \geq k$ is valid for kECSP(G), for $(i, j) \in (I_1 \times \{1\}) \cup (I_2 \times \{2\})$, we have

$$x([V_{2s-1}, V_{2s}]) + x([V_{2s}, V_{2s+1}]) \geq 1 \text{ for } s = 1, ..., p-1, \tag{2.5}$$

$$x([V_{2s}, V_{2s+1}]) + x([V_{2s+1}, V_{2s+2}]) \geq 1 \text{ for } s = 1, ..., p-1. \tag{2.6}$$

By multiplying each inequality (2.5) (resp. inequality (2.6)) corresponding to $s \in \{1, ..., p-1\}$ by $\frac{p-s}{p}$ (resp. $\frac{s}{p}$) and summing these inequalities, we obtain

$$\sum_{i \in I} x([V_i, V_{i+1}]) + \sum_{i \in \overline{I}} \frac{p-1}{p} x([V_i, V_{i+1}]) \geq p - 1, \tag{2.7}$$

where $I = \{2, 4, 6, ..., 2p-2\}$ and $\overline{I} = \{1, ..., 2p-1\} \setminus I$.

By considering the cut inequality induced by $W_1 \cup V_1 \cup (\bigcup_{i \in I_1} V_i)$ (resp. $W_1 \cup V_1 \cup (\bigcup_{i \in I_1} V_i) \cup V_{2p}$) if p is odd (resp. even) we have

$$x([W_1, W_2]) + \sum_{i \in \overline{I}} x([V_i, V_{i+1}]) \geq k.$$

As $|[W_1, W_2]| \leq k - 1$, it follows that

$$\frac{1}{p}\sum_{i\in\overline{I}} x([V_i, V_{i+1}]) \geq \frac{1}{p}. \tag{2.8}$$

By summing inequalities (2.7) and (2.8) and rounding up the right hand side, we get inequality (2.4). □

In what follows, we describe necessary conditions for inequality (2.4) to be facet defining. For this, we first give a technical lemma.

Lemma 2.2.3 *Let $\pi = (W_1, W_2, V_1, ..., V_{2p})$, $p \geq 2$, be a partition of V which induces an odd path configuration and F a solution of the kECSP. Let $V_r, ..., V_s$, with $2 \leq r < s \leq 2p-1$, be a sequence of node sets of π. Then F must contain at least $\lceil \frac{s-r+1}{2} \rceil$ edges from C.*

Proof. As $|[W_1, V_i]| = k-1$ for all $i \in \{r, ..., s\} \cap I_1$ and $|[W_2, V_i]| = k-1$ for all $i \in \{r, ..., s\} \cap I_2$, F must contain at least one edge from each set $\delta(V_i) \cap C$, $i \in \{r, ..., s\}$. Thus the statement follows. □

Theorem 2.2.2 *Inequality (2.4) defines a facet for kECSP(G) only if*

a) $[V_1, W_1] \neq \emptyset$ and $[V_{2p}, W_1] \neq \emptyset$ (resp. $[V_{2p}, W_2] \neq \emptyset$) if p is even (resp. odd),

b) $[V_i, V_{i+1}] \neq \emptyset$ for $i = 1, ..., 2p-1$.

Proof.

a) Suppose for instance that p is even and $[V_1, W_1] = \emptyset$ (the proof is similar if either $[V_{2p}, W_1] = \emptyset$ or p is odd and $[V_{2p}, W_2] = \emptyset$). By contracting the sets V_1, V_2, W_2, we obtain a smaller odd path configuration with $2p$ elements. Let

$$x(C') \geq p-1 \tag{2.9}$$

be the corresponding odd path inequality. As $\delta(V_2) = [V_1, V_2] \cup [V_2, V_3] \cup [V_2, W_2]$ and $|[V_2, W_2]| = k-1$, by the cut constraint on V_2, we have that

$$x([V_1, V_2]) + x([V_2, V_3]) \geq 1 \tag{2.10}$$

is valid for kECSP(G). By adding (2.9) and (2.10), we get $x(C) \geq p$, which implies that (2.4) cannot be facet defining.

b) Suppose that $[V_i, V_{i+1}] = \emptyset$ for some $i \in \{1, ..., 2p-1\}$. We will show in the following that any solution F of the kECSP whose the incidence vector x^F satisfies (2.4) with equality, intersects $[V_{i-1}, V_i]$ in exactly one edge. To this end, we will distinguish two cases.

Case 1. $i, i+1 \in I_1$ (the proof is similar if $i, i+1 \in I_2$). By Lemma 2.2.3 the edge set $F' = F \cap C$ must cover the node sets $V_2, ..., V_{i-2}$ by at least $\lceil \frac{i-3}{2} \rceil$ edges and the sets $V_{i+1}, ...V_{2p-1}$ by at least $\lceil \frac{2p-i-1}{2} \rceil$ edges. As $i, i+1 \in I_1$, and then i is even, F' must use, in consequence, at least $(\frac{i}{2} - 1) + (p - \frac{i}{2}) = p - 1$ edges from $C \setminus [V_{i-1}, V_i]$. Since $\delta(V_i) = [V_{i-1}, V_i] \cup [V_i, W_1]$ and $|[V_i, W_1]| = k - 1$, F contains at least one edge from $[V_{i-1}, V_i]$. As x^F satisfies (2.4) with equality, it follows that F contains exactly one edge from $[V_{i-1}, V_i]$.

Case 2. $i \in I_1$ and $i+1 \in I_2$ (the proof is similar if $i \in I_2$ and $i+1 \in I_1$). First note that in this case i is odd. By Lemma 2.2.3, F must cover the node sets $V_2, ..., V_{i-2}$ by at least $\lceil \frac{i-3}{2} \rceil = \frac{i-3}{2}$ edges from C and the node sets $V_{i+1}, ...V_{2p-1}$ by at least $\lceil \frac{2p-i-1}{2} \rceil = \frac{2p-i-1}{2}$ edges from C. Hence F uses at least $\frac{i-3}{2} + \frac{2p-i-1}{2} = p - 2$ edges from C. Moreover, observe that if exactly $p-2$ edges of C are used by F, then these edges should be between consecutive node sets of the form $[V_{2s}, V_{2s+1}]$, with $s \in \{1, ..., p-1\} \setminus \{\frac{i-1}{2}\}$. However, in this case, in order to satisfy the cut inequality induced by the node set $W_1 \cup (\bigcup_{r \in I_1} V_r) \cup V_{2p}$ (resp. $W_1 \cup (\bigcup_{r \in I_1} V_r)$) if p is even (resp. odd), F must contain at least one more edge from $C \setminus [V_{i-1}, V_i]$ between two consecutive sets of the form $[V_{2s-1}, V_{2s}]$, with $s \in \{1, ..., p-1\} \setminus \{\frac{i-1}{2}\}$. In consequence, F contains at least $p-1$ edges from $C \setminus [V_{i-1}, V_i]$. As $|F \cap [V_{i-1}, V_i]| \geq 1$ and x^F satisfies (2.4) with equality, we then have that $|F \cap [V_{i-1}, V_i]| = 1$.

In consequence, for any solution $F \subseteq E$ of the kECSP, if x^F satisfies (2.4) with equality, it also satisfies the equation $x(\delta(V_i)) = k$. Since kECSP(G) is full dimensionnal and (2.4) is not a positive multiple of $x(\delta(V_i)) \geq k$, (2.4) cannot define a facet. □

Now we give sufficient conditions for inequality (2.4) to be facet defining. For this, let us denote by Γ the set of edges of G which are not in C, that is, $\Gamma = E \setminus C$. Moreover, if $[V_i, V_{i+1}] \neq \emptyset$, we let e_i denote a fixed edge of $[V_i, V_{i+1}]$, for $i = 1, ..., 2p-1$.

Theorem 2.2.3 *Inequality (2.4) defines a facet for kECSP(G) if the following hold.*

i) Condition b) of Theorem 2.2.2 holds,

ii) The subgraphs $G[W_1]$, $G[W_2]$ and $G[V_i]$, for $i = 1, ..., 2p$, are $(k+1)$-edge connected,

iii) $|[W_1, W_2]| = k-1$, $|[V_1, W_1]| = k$ and $|[V_{2p}, W_1]| = k$ (resp. $|[V_{2p}, W_2]| = k$) if p is even (resp. odd).

Proof. We will show the result for p even (the proof is similar if p is odd).
Let $E_0 = \bigcup_{s=1}^{p} [V_{2s-1}, V_{2s}]$, $E_1 = \bigcup_{s=1}^{p-1} [V_{2s}, V_{2s+1}]$, $\overline{E} = \delta(\pi) \backslash (E_0 \cup E_1)$, $\tilde{E} = E \backslash (E_0 \cup E_1 \cup \overline{E})$.
Inequality (2.4) can then be written as

$$x(E_0) + x(E_1) \geq p. \tag{2.11}$$

Suppose that conditions 1) - 3) above hold. We first give a claim that will be useful in the proof.

Claim. If D is a subset of edges which covers the node sets $V_2, ..., V_{2p-1}$, contains at least one edge of $[V_{i_0}, V_{i_0+1}]$ for some $i_0 \in \{1, 3, ..., 2p-1\}$ and such that $D \cap \Gamma = \emptyset$, then $D \cup \Gamma$ induces a k-edge-connected subgraph of G.

Proof. Let $F = D \cup \Gamma$. Let $\overline{G}$ be the graph induced by F and $\overline{G}'$ the graph obtained from $\overline{G}$ by contracting the node sets $W_1, W_2, V_1, ..., V_{2p}$. Let $w_1, w_2, v_1, ..., v_{2p}$ be the nodes of $\overline{G}'$ where w_j (resp. v_i) corresponding to W_j (resp. V_i) for $j = 1, 2$ (resp. $i = 1, ..., 2p$). As by condition 2), the subgraphs of $\overline{G}$ induced by $W_1, W_2, V_1, ..., V_{2p}$ are $(k+1)$-edge connected, to show the claim, it suffices to show that $\overline{G}'$ is k-edge-connected. Let $\delta_{\overline{G}'}(W)$ be a cut of $\overline{G}'$.

If, say, $w_1 \in W$ and $w_2 \in \overline{W}$, then $[w_1, w_2] \subseteq \delta_{\overline{G}'}(W)$. If $\delta_{\overline{G}'}(W)$ separates v_{i_0} and v_{i_0+1}, as D intersects $[V_{i_0}, V_{i_0+1}]$, and by condition 3), $|[W_1, W_2]| = k-1$, we have that $|\delta_{\overline{G}'}(W)| \geq k$. If $v_{i_0}, v_{i_0+1} \in W$, then $[\{v_{i_0}, v_{i_0+1}\}, w_2] \subseteq \delta_{\overline{G}'}(W)$. Since $|[\{v_{i_0}, v_{i_0+1}\}, w_2]| \geq k-1 \geq 1$, this yields $|\delta_{\overline{G}'}(W)| \geq k$.

Now if $w_1, w_2 \in W$ (or $w_1, w_2 \in \overline{W}$), then $\delta_{\overline{G}'}(W)$ contains at least two edge sets of the form $[v_i, w_j]$ with $(i, j) \in (I_1 \times \{1\}) \cup (I_2 \times \{2\})$. Since $|[v_i, w_j]| = k-1$, we have that $|\delta_{\overline{G}'}(W)| \geq k$.

♦

Let us denote inequality (2.11) by $ax \geq \alpha$ and $\mathcal{F} = \{x \in k\text{ECSP}(G) \mid ax = \alpha\}$. Let $S = \Gamma \cup \{e_{2s-1},\ s = 1, ..., p\}$. By the claim above, we can see that S induces a k-edge-connected subgraph of G. Moreover, x^S satisfies (2.11) with equality, which implies that $\mathcal{F}$ is a proper face of $k\text{ECSP}(G)$. Now suppose that there exists a non trivial facet defining inequality $bx \geq \beta$ such that $\mathcal{F} \subseteq \{x \in k\text{ECSP}(G) \mid bx = \beta\}$. By Lemma 2.2.1, we have that $\beta > 0$, and hence we may suppose that $\beta = \alpha$. As G is $(k+1)$-edge connected and thus $k\text{ECSP}(G)$ is full dimensional, it suffices to show that $b = a$.

Let $e \in [V_{2s-1}, V_{2s}] \setminus \{e_{2s-1}\}$ for some $s \in \{1, ..., p\}$ and $S_1 = (S \setminus \{e_{2s-1}\}) \cup \{e\}$. By the claim above, S_1 induces a k-edge-connected subgraph of G. Moreover, $ax^{S_1} = \alpha$. It then follows that $bx^{S_1} = \alpha$, implying that

$$b(e) = \rho_{2s-1} \text{ for all } e \in [V_{2s-1}, V_{2s}],\ \text{for } s = 1, ..., p,\ \text{for some } \rho_{2s-1} \in \mathbb{R},\ \rho_{2s-1} \neq 0. \tag{2.12}$$

Similarly, for an edge $e \in [V_{2s}, V_{2s+1}] \setminus \{e_{2s}\}$ for some $s \in \{1, ..., p-1\}$ one can consider the edge sets $S_2 = \Gamma \cup (\bigcup_{i=1}^{p-1} \{e_{2i}\}) \cup \{e_1\}$ and $S_3 = (S_2 \setminus \{e_{2s}\}) \cup \{e\}$. We can see by the claim above that S_2 and S_3 induce k-edge-connected subgraphs of G. Since, $ax^{S_2} = ax^{S_3} = \alpha$, it follows that $bx^{S_2} = bx^{S_3} = \alpha$ and then

$$b(e) = \rho_{2s} \text{ for all } e \in [V_{2s}, V_{2s+1}],\ \text{for } s = 1, ..., p-1,\ \text{for some } \rho_{2s} \in \mathbb{R},\ \rho_{2s} \neq 0. \tag{2.13}$$

Consider the edge sets $S_4 = (S_2 \setminus \{e_1\}) \cup \{e_{2s-1}\}$ and $S_5 = (S_2 \setminus \{e_1, e_{2s}\}) \cup \{e_{2s-1}, e_{2s+1}\}$ for some $s \in \{1, ..., p-1\}$. By the claim above, S_4 and S_5 induce k-edge-connected subgraphs of G. Since $ax^{S_4} = ax^{S_5} = \alpha$, $bx^{S_4} = bx^{S_5} = \alpha$ and hence

$$b(e_1) = b(e_{2s}) = b(e_{2s+1}),\ \text{for } s = 1, ..., p-1. \tag{2.14}$$

From (2.12), (2.13) and (2.14), it follows that $b(e)$ is the same for every edge $e \in E_0 \cup E_1$. Since $ax^S = bx^S = \alpha$, we get $b(e) = 1$ for all $e \in E_0 \cup E_1$.

Now we are going to show that $b(e) = 0$ for all $e \in \tilde{E} \cup \overline{E}$. For this, first consider an edge $f \in \tilde{E}$. From condition 2), $S_f = S \setminus \{f\}$ induces a k-edge-connected subgraph of

G. Moreover, x^{S_f} satisfies (2.11) with equality. Hence $ax^{S_f} = \alpha = bx^{S_f}$. This implies that $b(f) = bx^S - bx^{S_f} = 0$.

Now let $e \in [V_i, W_j]$ for $(i,j) \in (I_1 \cup \{1, 2p\} \times \{1\}) \cup (I_2 \times \{2\})$ and $S_6 = (S_2 \setminus \{e_1\}) \cup \{e_{i-1}\}$ (resp. $S_6 = (S_2 \setminus \{e_1\}) \cup \{e_i\}$) if i is even (resp. odd). From the claim above, we have that S_6 and $S'_6 = S_6 \setminus \{e\}$ induce k-edge-connected subgraphs of G and that their incidence vectors satisfy $ax \geq \alpha$ with equality. Hence $b(e) = bx^{S_6} - bx^{S'_6} = 0$.

For all $e \in [W_1, W_2]$, by the claim above, the edge set $S_7 = S \setminus \{e\}$ induces a k-edge-connected subgraph of G. Moreover, x^{S_7} satisfies $ax \geq \alpha$ with equality. Hence $ax^{S_7} = \alpha$ and $bx^{S_7} = bx^S = \alpha$. Thus we obtain $b(e) = 0$ for all $e \in [W_1, W_2]$.

Consequently, $b(e) = 0$ for all $e \in E \setminus C$, which terminates the proof of the theorem. □

2.2.2 Lifting procedure for odd path inequalities

In what follows we are going to describe a lifting procedure for the odd path inequalities. This will permit to extend these inequalities to a more general class of valid inequalities. But first we give the following lemma which easily follows from the general lifting procedure presented in [93].

Lemma 2.2.4 *Let $G = (V, E)$ be a graph and $ax \geq \alpha$ a valid inequality for kECSP(G). Let $G' = (V, E')$ be a graph obtained from G by adding an edge e, that is $E' = E \cup \{e\}$. Then the inequality*

$$ax + a(e)x(e) \geq \alpha \tag{2.15}$$

is valid for kECSP(G') where $a(e) = \alpha - \gamma$ with $\gamma = min\{ax \mid x \in k\text{ECSP}(G') \text{ and } x(e) = 1\}$. Moreover, if $ax \geq \alpha$ is facet defining for kECSP(G), then inequality (2.15) is also facet defining for kECSP(G'). In addition, if edges $e_1, ..., e_{k-1}, e_k, ..., e_t$ are added to G in this order and $a(e_k)$ is the lifting coefficient of e_k with respect to this order, then $a(e_k) \leq a'(e_k)$ where $a'(e_k)$ is the lifting coefficient of e_k in any order $e_{i_1}, ..., e_{i_{k-1}}, ..., e_{i_t}$ such that $i_l = l$ for $l = 1, ..., k-1$ and $i_s = k$ for some $s \geq k$.

Theorem 2.2.4 *Let $G = (V, E)$ be a graph and $\pi = (W_1, W_2, V_1, ..., V_{2p})$, $p \geq 2$, a partition of V which induces an odd path configuration. Let C, I_1 and I_2 be defined as in Section 2.2.1. Let $U_1 = \bigcup_{i \in I_1} V_i$, $U_2 = \bigcup_{i \in I_2} V_i$ and $W = U_2 \cup V_{2p} \cup W_2$ (resp. $W = U_2 \cup W_2$) if p is odd (resp. even). Suppose that conditions 1) - 3) of Theorem*

2.2.3 hold. If $G' = (V, E \cup L)$ is a graph obtained from G by adding an edge set L, then the following inequality

$$x(C) + \sum_{e \in L} a(e)x(e) \geq p, \tag{2.16}$$

with

$$a(e) = \begin{cases} 1 \text{ if } e \in (\bigcup_{j=1,2} [W_j, U_1 \cup U_2]) \cup [W_1, W_2] \cup (\bigcup_{j=1,2p} [V_j, U_1 \cup U_2]) \text{ or} \\ \quad e \in ([V_1, V_{2p} \cup W_2] \cup [V_{2p}, W_1 \cup W_2]) \cap \delta(W), \\ 2 \text{ if } e \in [V_i, V_j],\ i, j \in \{2, ..., 2p-1\} \text{ with } j > i+1 \text{ and } i \text{ even, } j \text{ odd}, \\ \lambda \text{ if } e \in [V_i, V_j] \text{ with } i, j \in \{2, ..., 2p-1\},\ j > i+1 \text{ and } i \text{ odd} \\ \quad \text{or } i \text{ and } j \text{ have same parity}, \\ 0 \text{ otherwise}, \end{cases}$$

where $1 \leq \lambda \leq 2$ is the lifting coefficient obtained using the lifting procedure of Lemma 2.2.4, is facet defining for kECSP(G').

Proof. Let us consider the following edge subsets of L:

$$L_1 = (\bigcup_{j=1,2} [W_j, U_1 \cup U_2]) \cup [W_1, W_2] \cup (\bigcup_{j=1,2p} [V_j, U_1 \cup U_2]) \cup (([V_1, V_{2p} \cup W_2] \cup [V_{2p}, W_1 \cup W_2]) \cap \delta(W)),$$

$$L_2 = \{[V_i, V_j],\ i, j \in \{2, ..., 2p-1\},\ j > i+1,\ i \text{ even },\ j \text{ odd}\},$$

$$L_3 = \{[V_i, V_j],\ i, j \in \{2, ..., 2p-1\},\ j > i+1,\ i \text{ odd or, } i \text{ and } j \text{ have the same parity}\},$$

$$L_4 = L \setminus (L_1 \cup L_2 \cup L_3).$$

We will first show that the lifting coefficient of the edges of L_4 is equal to 0, independently of the order in which they are added to G. Let e be an edge of L_4 and let us denote by $a'x \geq \alpha'$ the lifted inequality obtained on G'. As, by our assumptions, (2.4) defines a facet of kECSP(G), $a'x \geq \alpha'$ also defines a facet of kECSP(G'). Since $a'x \geq \alpha'$ is different from the trivial inequality $x(e) \geq 0$, there must exist a solution $F' \subseteq E'$ of the kECSP on G' such that $e \in F'$ and whose the incidence vector satisfies

$a'x \geq \alpha'$ with equality. Let $h_1, ..., h_k$ be the edges of E between V_1 and W_1. Note that $a'(h_1) = ... = a'(h_k) = 0$. We will distinguish two cases.

Case 1. $|[F' \cap \{h_1, ..., h_k\}]| \leq k - 1$. Let h_i be an edge not contained in F'. Let $F'' = (F' \setminus \{e\}) \cup \{h_i\}$. Since F' induces a k-edge-connected subgraph of G', F'' so is. Hence we have that $a'x^{F''} = a'x^{F'} - a'(e) + a'(h_i) \geq \alpha'$. This yields $a'(e) \leq a'(h_i)$. Since $a'(h_i) = 0$, and by Lemma 2.2.1, $a'(e) \geq 0$, we get $a'(e) = 0$.

Case 2. $\{h_1, ..., h_k\} \subseteq F'$. Here we also have that $F'' = F' \setminus \{e\}$ induces a k-edge-connected subgraph of G'. As $a'x^{F''} = a'x^{F'} - a'(e) \geq \alpha'$, and thus $a'(e) \leq 0$, it follows, by Lemma 2.2.1, that $a'(e) = 0$.

Therefore $a(e) = 0$ for all $e \in L_4$, and this, independently of the order in which e is added to G.

Now we consider the edges of $L \setminus L_4$. For this, we give the following claim.

Claim. $a(e) \geq 1$ if $e \in L_1 \cup L_3$, and $a(e) \geq 2$ if $e \in L_2$.

Proof. We will show first that if we add one edge $e \in L_1$ (resp. $e \in L_2$) (resp. $e \in L_3$) to G, the lifting coefficient of e in the new graph is 1 (resp. 2) (resp. 1). For this, let us denote by $\tilde{G} = (V, \tilde{E})$ the graph obtained by adding the edge e, that is, $\tilde{E} = E \cup \{e\}$. Suppose first that $e \in L_1$ and assume that, for instance, $e \in [W_{j_0}, V_{i_0}]$, with $i_0 \in \{2, ..., 2p-1\}$ and even, and $j_0 \in \{1, 2\}$ (if i_0 is odd, it suffices to consider the path $V_1, ..., V_{2p}$ in the opposite way). Note that any solution $\tilde{F} \subseteq \tilde{E}$ of the kECSP on $\tilde{G}$ must cover the node sets $V_2, ..., V_{i_0-1}$ and $V_{i_0+1}, ..., V_{2p-1}$ by edges from C. By Lemma 2.2.3, $\tilde{F}$ must use at least $\lceil \frac{i_0-2}{2} \rceil + \lceil \frac{2p-i_0-1}{2} \rceil = p - 1$ edges from C. Thus $\gamma \geq p - 1$ where γ is as defined in Lemma 2.2.4. Moreover, because the conditions of Theorem 2.2.3 are satisfied, by the claim given in the proof of that theorem, the edge set $\tilde{F}_1 = \{e_2, e_4, ..., e_{i_0-2}\} \cup \{e_{i_0+1}, e_{i_0+3}, ..., e_{2p-1}\} \cup \Gamma \cup \{e\}$ induces a k-edge-connected subgraph of $\tilde{G}$. Since $\tilde{F}_1$ contains e and uses exactly $p - 1$ edges from C, we have that $\gamma = p - 1$. By Lemma 2.2.4, it then follows that the lifting coefficient of e is equal to 1.

Consider now an edge $e \in L_2$ and suppose that $e \in [V_{i_0}, V_{j_0}]$ with $i_0, j_0 \in \{2, ..., 2p-1\}$, $j_0 > i_0+1$, and i_0 is even and j_0 odd. If $\tilde{F}$ is a solution of the kECSP on $\tilde{G}$, then $\tilde{F}$ must cover the node sets $V_2, ..., V_{i_0-1}$, $V_{i_0+1}, ..., V_{j_0-1}$ and $V_{j_0+1}, ..., V_{2p-1}$. Thus by Lemma 2.2.3, $\tilde{F}$ must use $\lceil \frac{i_0-2}{2} \rceil + \lceil \frac{j_0-i_0-1}{2} \rceil + \lceil \frac{2p-j_0-1}{2} \rceil = p-2$ edges from C. Thus, $\gamma \geq p-2$. Now let $\tilde{F}_2 = \{e_2, e_4, ..., e_{i_0-2}\} \cup \{e_{i_0+1}, e_{i_0+3}, ..., e_{j_0-2}\} \cup \{e_{j_0+1}, e_{j_0+3}, ..., e_{2p-2}\} \cup \Gamma \cup \{e\}$. We can see as before that $\tilde{F}_2$ induces a k-edge-connected subgraph of $\tilde{G}$ and contains exactly $p-2$ edges from C. Since $e \in \tilde{F}_2$, we obtain that $\gamma = p-2$, and therefore the lifting coefficient of e equals 2.

Finally, suppose that e is an edge of L_3 between two non consecutive node sets $[V_{i_0}, V_{j_0}]$ with $i_0, j_0 \in \{2, ..., 2p-1\}$, $j_0 > i_0 + 1$, and , say, i_0 is odd and j_0 is even (the proof is similar if i_0 and j_0 have the same parity). Here observe that any solution $\tilde{F} \subseteq \tilde{E}$ of the kECSP on $\tilde{G}$ must cover by edges from C the node sets $V_2, ...V_{i_0-1}$, $V_{i_0+1}, ..., V_{j_0-1}$ and $V_{j_0+1}, ..., V_{2p-1}$. By Lemma 2.2.3, $\tilde{F}$ must then use at least $\lceil \frac{i_0-2}{2} \rceil + \lceil \frac{j_0-i_0-1}{2} \rceil + \lceil \frac{2p-j_0-1}{2} \rceil = p-1$ edges from C. Thus $\gamma \geq p-1$. Moreover, as the edge set $\tilde{F}_3 = \{e_1, e_3, ..., e_{i_0-2}\} \cup \{e_{i_0+1}, e_{i_0+1}, ..., e_{2p-2}\} \cup \Gamma \cup \{e\}$ induces a k-edge-connected subgraph of $\tilde{G}$ and contains exactly $p-1$ edges from C, we have that $\gamma = p-1$. Hence the lifting coefficient of e in $\tilde{G}$ is equal to 1.

Consequently the lifting coefficient of e equals 1 (resp. 2) (resp. 1) if $e \in L_1$ (resp. $e \in L_2$) (resp. $e \in L_3$). By Lemma 2.2.4, we then have that $a(e) \geq 1$ if $e \in L_1 \cup L_3$ and $a(e) \geq 2$ if $e \in L_2$, which ends the proof of the claim. ♦

In what follows, we are going to show that we also have $a(e) \leq 1$ (resp. $a(e) \leq 2$) (resp. $1 \leq a(e) \leq 2$) if $e \in L_1$ (resp. $e \in L_2$) (resp. $e \in L_3$). For this, let us consider a sequence $f_1, ..., f_t$, $t \geq 1$, of edges of L, and suppose that $f_1, ..., f_t$ are the edges that are added to G before e.

Suppose first that $e \in L_1$ and let us assume as before that $e \in [W_{j_0}, V_{i_0}]$ with $i_0 \in \{2, ..., 2p-1\}$ and even, and $j_0 \in \{1, 2\}$. Let $\hat{G} = (V, \hat{E})$ be the graph where $\hat{E} = E \cup \{f_1, ..., f_t, e\}$. Any solution $\hat{F} \subseteq \hat{E}$ of the kECSP on $\hat{G}$ must cover the node sets $V_2, ..., V_{i_0-1}$ and $V_{i_0+1}, ..., V_{2p-1}$ by edges from $(C \cup \{f_1, ..., f_t\}) \setminus L_4$. By Lemma 2.2.3, $\hat{F}$ must use at least $\lceil \frac{i_0-2}{2} \rceil + \lceil \frac{2p-i_0-1}{2} \rceil = p-1$ edges from $(C \cup \{f_1, ..., f_t\}) \setminus L_4$. Since, by the claim above, $a(f) \geq 1$ for every edge $f \in (C \cup \{f_1, ..., f_t\}) \setminus L_4$, we have that $\gamma \geq p-1$ and hence by Lemma 2.2.4, we have that $a(e) \leq 1$. As, by the claim above $a(e) \geq 1$, this implies that $a(e) = 1$. Moreover, this holds independently on the order in which e is added to G.

Now consider an edge $e \in L_2$ and suppose that $e \in [V_{i_0}, V_{j_0}]$, with $i_0, j_0 \in \{2, ..., 2p-1\}$, $j_0 > i_0 + 1$, i_0 even and j_0 odd. Any solution $\hat{F} \subseteq \hat{E}$ of the kECSP on $\hat{G}$ must cover the node sets $V_2, ..., V_{i_0-1}$, $V_{i_0+1}, ..., V_{j_0-1}$ and $V_{j_0+1}, ..., V_{2p-1}$ by edges from $(C \cup \{f_1, ..., f_t\}) \setminus L_4$. By Lemma 2.2.3, $\hat{F}$ must use $\lceil \frac{i_0-2}{2} \rceil + \lceil \frac{j_0-i_0-1}{2} \rceil + \lceil \frac{2p-j_0-1}{2} \rceil = p-1$ edges of $(C \cup \{f_1, ..., f_t\}) \setminus L_4$. Thus $\gamma \geq p-2$ and therefore $a(e) \leq 2$. Since, by the claim above, $a(e) \geq 2$, we get $a(e) = 2$.

If e is an edge of L_3, we show along the same line that $1 \leq a(e) \leq 2$.

In consequence, $a(e) = 1$ if $e \in L_1$, $a(e) = 2$ if $e \in L_2$, $1 \leq a(e) \leq 2$, which ends the proof of the theorem.

□

Observe that the lifting coefficients of the edges other than those between two subsets V_i and V_j such that $i, j \in \{2, ..., 2p-1\}$, $j > i+1$, i is odd or i and j have the same parity do not depend on the order of their addition in G. Inequalities (2.16) will be called *lifted odd path inequalities*. As it will turn out, these inequalities are very useful for our Branch-and-Cut algorithm.

2.2.3 F-partition inequalities

In [88], Mahjoub introduced a class of valid inequalities for 2ECSP(G) as follows. Let $(V_0, V_1, ..., V_p)$, $p \geq 2$, be a partition of V and $F \subseteq \delta(V_0)$ with $|F|$ odd. By adding the inequalities

$$x(\delta(V_i)) \geq 2 \quad \text{for } i = 1, ..., p, \tag{2.17}$$

$$-x(e) \geq -1 \quad \text{for } e \in F, \tag{2.18}$$

$$x(e) \geq 0 \quad \text{for } e \in \delta(V_0) \setminus F, \tag{2.19}$$

we obtain $2x(\Delta) \geq 2p - |F|$ where $\Delta = \delta(V_0, V_1, ..., V_p) \setminus F$. Dividing by 2 and rounding up the right hand side lead to

$$x(\Delta) \geq p - \frac{|F|-1}{2}. \tag{2.20}$$

Inequalities (2.20) are called *F-partition inequalities*. Didi Biha [38] extended these inequalities for all $k \geq 2$. He showed that, given a partition $(V_0, V_1, ..., V_p)$, $p \geq 2$, of V and $F \subseteq \delta(V_0)$ with $F \neq \emptyset$, the inequality

$$x(\delta(V_0, V_1, ..., V_p) \setminus F) \geq \left\lceil \frac{kp - |F|}{2} \right\rceil, \tag{2.21}$$

is valid for kECSP(G). Note here that $|F|$ can be either odd or even. Also note that if kp and $|F|$ have the same parity, then the corresponding inequality (2.21) is implied by the cut and the trivial inequalities.

In what follows, we describe sufficient conditions for inequalities (2.21) to be facet defining. Theorems 2.2.5 and 2.2.6 describe these conditions for k odd and k even, respectively. Note that all the indices we will consider here will be modulo $2l + 1$.

Theorem 2.2.5 *Let $G = (V, E)$ be a graph and $k \geq 3$ an odd integer. Let $\pi = (W, V_1, ..., V_{2l+1},$
$U_1, ..., U_{2l+1})$, with $l \geq \frac{k-1}{2}$, be a partition of V such that*

i) $G[W]$, $G[V_i]$, $G[U_i]$, $i = 1, ..., 2l + 1$, *are $(k + 1)$-edge connected,*

ii) $|[W, V_i]| \geq k - 2$ *for* $i = 1, ..., 2l + 1$,

iii) $|[U_i, U_{i+1}]| \geq \frac{k-1}{2}$ *for* $i = 1, ..., 2l + 1$,

iv) $|[V_i, V_{i+1}]| \geq 1$ *for* $i = 1, ..., 2l + 1$,

v) $|[V_i, U_i]| \geq 1$ *and* $|[V_i, U_{i-1}]| \geq 1$ *for* $i = 1, ..., 2l + 1$
(see Figure 2.2 for an illustration with $k = 5$ and $l = 2$).

Let F_i be an edge subset of $[W, V_i]$ such that $|F_i| = k - 2$, $i = 1, ..., 2l + 1$ and let $F = \bigcup_{i=1}^{2l+1} F_i$. Then the F-partition inequality

$$x(\delta(\pi) \setminus F) \geq l(k + 2) + \left\lceil \frac{k}{2} \right\rceil + 1, \tag{2.22}$$

induced by π and F, defines a facet of kECSP(G).

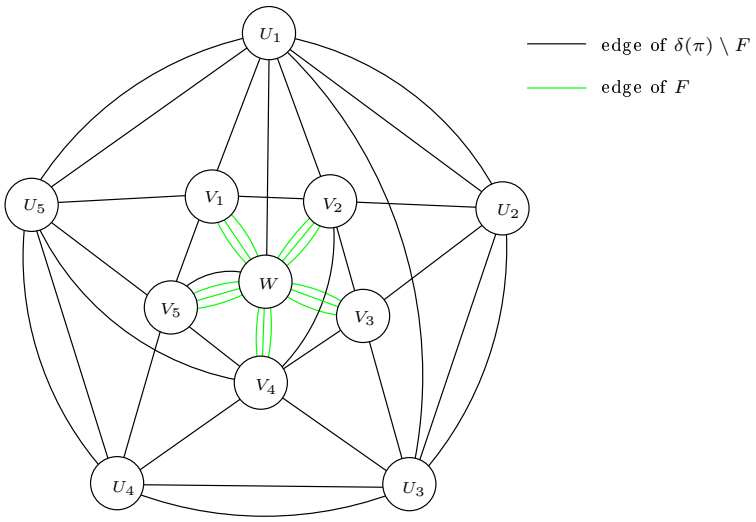


Figure 2.2: An F-partition configuration with $k = 5$

Proof. First observe that, by conditions 1) - 5), G is $(k+1)$-edge connected and hence kECSP(G) is full dimensional. Let us denote inequality (2.22) by $ax \geq \alpha$ and let $\mathcal{F} = \{x \in k\text{ECSP}(G) \mid ax = \alpha\}$. Clearly, $\mathcal{F}$ is a proper face of kECSP(G). Now suppose that there exists a facet defining inequality $bx \geq \alpha$ such that $\mathcal{F} \subseteq \{x \in k\text{ECSP}(G) \mid bx = \alpha\}$. We will show that $b = a$.

Let e_i be an edge of $[V_i, V_{i+1}]$, $i = 1, ..., 2l+1$, and f_i and f_i' be edges of $[V_i, U_{i-1}]$ and $[V_i, U_i]$, respectively, for $i = 1, ..., 2l+1$. Let T_i be an edge subset of $[U_i, U_{i+1}]$ of $\frac{k-1}{2}$ edges, for $i = 1, ..., 2l+1$.

Let E_0 be the set of edges not in F and having both endnodes in the same element of π. First we will show that $b(e) = 0$ for all $e \in E_0 \cup F$. Let $i_0 \in \{1, ..., 2l+1\}$ and consider the edge sets

$$E_1 = \{e_{i_0+2r},\ r = 0, ..., l\} \cup \{f_i',\ i = 1, ..., 2l+1\} \cup (\bigcup_{i=1}^{2l+1} T_i),$$

$$E_2 = E_1 \cup F \cup E_0.$$

Claim. E_2 induces a k-edge-connected subgraph of G.

Proof. Let G_2 be the subgraph of G induced by E_2. Since by condition 1) the graphs induced by the node sets W and V_i, U_i, $i = 1, ..., 2l+1$, are $(k+1)$-edge connected, it suffices to show that the graph obtained by contracting W and V_i, U_i, $i = 1, ..., 2l+1$, is

k-edge-connected. Let $\overline{G}_2 = (\overline{V}_2, \overline{E}_2)$ be that graph and w ,$v_1, ..., v_{2l+1}$, $u_1, ..., u_{2l+1}$ the nodes of $\overline{G}_2$ where w corresponds to W, v_i to V_i and u_i to U_i, for $i = 1, ..., 2l+1$. Let $\delta(U)$ be a cut of $\overline{G}_2$ and let $\overline{G}'_2 = (\overline{V}'_2, \overline{E}'_2)$ the subgraph of $\overline{G}_2$ induced by $\{w, v_1, ..., v_{2l+1}\}$ and $\overline{G}''_2 = (\overline{V}''_2, \overline{E}''_2)$ the graph obtained from $\overline{G}_2$ by contracting $\{w, v_1, ..., v_{2l+1}\}$. Note that $\overline{E}'_2 \cap \overline{E}''_2 = \emptyset$ and $\overline{E}_2 = \overline{E}'_2 \cup \overline{E}''_2$. Also note that $\overline{G}'_2$ is $(k-1)$-edge connected and that $\overline{G}''_2$ is a k-edge-connected wheel. Thus if U does not intersect $\{w, v_1, ..., v_{2l+1}\}$, then $\delta(U)$ is a cut of $\overline{G}''_2$ and hence $|\delta(U)| \geq k$. If U intersects $\{w, v_1, ..., v_{2l+1}\}$, then $\delta(U)$ contains at least $k-1$ edges from $\overline{E}'_2$. However, in this case $\delta(U)$ also contains at least one edge from $\overline{E}''_2$. Thus we have that $|\delta(U)| \geq k$, and the statement follows. ♦

Note that there are $k+1$ edges incident to V_{i_0} in the graph induced by E_2. Now, observe that for any edge $e \in F_{i_0}$, one can show in a similar way as in the claim above that $E'_2 = E_2 \setminus \{e\}$ also induces a k-edge-connected subgraph of G. As x^{E_2} and $x^{E'_2}$ belong to $\mathcal{F}$, it follows that $bx^{E_2} = bx^{E'_2} = \alpha$, implying that $b(e) = 0$ for all $e \in F_{i_0}$. As i_0 is arbitrarily chosen, we obtain that $b(e) = 0$ for all $e \in F$. Moreover, as the subgraphs induced by W, $V_1, ..., V_{2l+1}$, $U_1, ..., U_{2l+1}$ are all $(k+1)$-edge connected, the subgraph induced by $E_2 \setminus \{e\}$, for all $e \in E_0$, is also k-edge-connected. This yields as before $b(e) = 0$ for all $e \in E_0$. Thus $b(e) = 0$ for all $e \in F \cup E_0$.

Next, we will show that $b(e) = a(e)$ for all $e \in \delta(\pi) \setminus F$. Let g_i be a fixed edge of T_i and let $T'_i = T_i \setminus \{g_i\}$, for $i = 1, ..., 2l+1$. Consider the edge sets

$$
\begin{aligned}
E_3 &= \{f_i, f'_i,\ i = 1, ..., 2l+1\} \cup (\bigcup_{i=1}^{l} T_{2i}) \cup T_{2l+1} \cup (\bigcup_{i=0}^{l-1} T'_{2i+1}),\\
E_4 &= E_3 \cup F \cup E_0,\\
E'_4 &= (E_4 \setminus g_{2l+1}) \cup \{g_1\}.
\end{aligned}
$$

Note that $g_1 \notin T'_1$ and thus $g_1 \notin E_4$, and that $g_{2l+1} \in E_4$. The edge sets E_4 and E'_4 can be obtained from E_2 using recursively the edge-swapping operation of Lemma 2.2.2. Hence both E_4 and E'_4 induce k-edge-connected subgraphs of G. Moreover, we have that x^{E_4} and $x^{E'_4}$ belong to $\mathcal{F}$. Thus $bx^{E_4} = bx^{E'_4} = \alpha$ and therefore $b(g_{2l+1}) = b(g_1)$. As g_1 and g_{2l+1} are arbitrary edges of T_1 and T_{2l+1}, respectively, it follows that $b(e) = b(e')$ for all $e \in T_1$ and $e' \in T_{2l+1}$. Moreover, we have that T_1 and T_{2l+1} are arbitrary subsets of $[U_1, U_2]$ and $[U_{2l+1}, U_1]$, respectively. This implies that $b(e) = b(e')$ for all $e \in [U_1, U_2]$ and $e' \in [U_{2l+1}, U_1]$. Consequently, by symmetry, we get

$$
\begin{aligned}
&b(e) = \rho \text{ for all } e \in [U_i, U_{i+1}],\ i = 1, ..., 2l+1,\\
&\text{for some } \rho \in \mathbb{R}.
\end{aligned}
\tag{2.23}
$$

Now let

$$E_5 = (E_4 \setminus \{f_1\}) \cup \{e_{2l+1}\}.$$

Using Lemma 2.2.2 and the fact that E_4 induces a k-edge-connected subgraph of G, we have that E_5 induces a k-edge-connected subgraph of G. Moreover, x^{E_5} belongs to $\mathcal{F}$, implying that $bx^{E_4} = bx^{E_5} = \alpha$. Hence $b(f_1) = b(e_{2l+1})$. In a similar way, we can show that $b(f'_{2l+1}) = b(e_{2l+1})$. As f_1, f'_{2l+1} and e_{2l+1} are arbitrary edges of $[U_{2l+1}, V_1]$, $[V_{2l+1}, U_{2l+1}]$ and $[V_{2l+1}, V_1]$, respectively, we obtain that $b(e)$ is the same for all $e \in [U_{2l+1}, V_1] \cup [V_{2l+1}, U_{2l+1}] \cup [V_{2l+1}, V_1]$. By exchanging the roles of V_{2l+1}, V_1, U_{2l+1} and V_i, V_{i+1}, U_i, for $i = 1, ..., 2l$, we obtain by symmetry that

$$\begin{aligned} &b(e) = \rho'_i \text{ for all } e \in [U_i, V_i] \cup [V_i, V_{i+1}] \cup [V_{i+1}, U_i], \\ &i = 1, ..., 2l+1, \text{for some } \rho'_i \in \mathbb{R}. \end{aligned} \tag{2.24}$$

Consider the edge set

$$E'_5 = (E_4 \setminus \{f_1\}) \cup \{e_1\}.$$

Similarly, we can show that E'_5 induces a k-edge-connected subgraph of G. As x^{E_4} and $x^{E'_5}$ belong to $\mathcal{F}$, it follows in a similar way that $b(e_1) = b(f_1)$. From (2.24), we have that $\rho'_1 = \rho'_{2l+1}$. By symmetry, it then follows that $\rho'_i = \rho'_j$ for $i, j = 1, ..., 2l+1$, $i \neq j$, and therefore

$$\begin{aligned} &b(e) = \rho' \text{ for all } e \in [U_i, V_i] \cup [V_i, V_{i+1}] \cup [V_{i+1}, U_i], \\ &\text{for } i = 1, ..., 2l+1, \text{ for some } \rho' \in \mathbb{R}. \end{aligned} \tag{2.25}$$

Let $e \in ([V_{2l+1}, W] \setminus F_{2l+1}) \cup [V_{2l+1}, V_j]$, $j \in \{2, ..., 2l-1\}$. As before, we can observe that $E_6 = (E_4 \setminus \{f'_{2l+1}\}) \cup \{e\}$ induces a k-edge-connected subgraph of G. Since $x^{E_6} \in \mathcal{F}$, this implies that $bx^{E_6} = bx^{E_4} = \alpha$ and hence $b(e) = b(f'_{2l+1})$. By (2.25), we then obtain that $b(e) = \rho'$ for all $e \in ([V_{2l+1}, W] \setminus F_{2l+1}) \cup [V_{2l+1}, V_i]$ for $i \in \{2, ..., 2l-1\}$. By exchanging the roles of V_{2l+1} and V_i, $i = 1, ..., 2l$, we obtain by symmetry that $b(e) = \rho'$ for all $e \in ([V_i, W] \setminus F_i) \cup [V_i, V_j]$, $i = 1, ..., 2l+1$ and $j \in \{1, ..., 2l+1\} \setminus \{i-1, i, i+1\}$.

For any edge e between U_{2l+1} and either W, U_j, $j \in \{1, ..., 2l+1\} \setminus \{1, 2l, 2l+1\}$, or V_t, $t \in \{1, ..., 2l+1\} \setminus \{1, 2l+1\}$, we can show, using Lemma 2.2.2 and the fact that E_4 induces a k-edge-connected subgraph of G, that

$$E_7 = (E_4 \setminus \{f'_{2l+1}, f_1\}) \cup \{e, e_{2l+1}\}$$

also induces a k-edge-connected subgraph of G. Since x^{E_4} and x^{E_7} belong to $\mathcal{F}$, we have that $bx^{E_7} = bx^{E_4} = \alpha$ and $b(f'_{2l+1}) + b(f_1) = b(e) + b(e_{2l+1})$. As by (2.25),

$b(f'_{2l+1}) = b(f_1) = b(e_{2l+1}) = \rho'$, we get $b(e) = \rho'$. Here again, by exchanging the roles of U_{2l+1} and U_i, $i = 1, ..., 2l$, we obtain that $b(e) = \rho'$ for all $e \in [U_i, W] \cup [U_i, U_j] \cup [U_i, V_t]$, $i = 1, ..., 2l+1$, $j \in \{1, ..., 2l+1\} \setminus \{i, i+1\}$ and $t \in \{1, ..., 2l+1\} \setminus \{i-1, i, i+1\}$.

As x^{E_2} and x^{E_4} belong to $\mathcal{F}$, we have that $bx^{E_2} = bx^{E_4} = \alpha$. Thus from (2.23) and (2.25), we obtain that $\rho = \rho'$, and in consequence, the edges of $E \setminus (E_0 \cup F)$ have all the same coefficient in $bx \geq \alpha$. Since $ax^{E_2} = bx^{E_2} = \alpha$, this yields $b(e) = 1$ for all $e \in E \setminus (E_0 \cup F)$.

Thus we obtain that $b = a$, which ends the proof of the theorem. □

We now describe special cases in which inequalities (2.21) define facets when k is even. Consider a graph $G = (V, E)$ and an even integer $k = 2q$ with $q \geq 1$, a *generalized odd-wheel configuration* is given by an integer $l \geq 1$, a set of positive integers $\{p_1, ..., p_{2l+1}\}$ and a partition $\pi = (V_0, V_i^s, i = 1, ..., 2l+1, s = 0, ..., p_i)$ of V such that

i) $G[V_0]$ and $G[V_i^s]$ are $(k+1)$-edge connected, for $s = 1, ..., p_i$ and $i = 1, ..., 2l+1$,

ii) $|[V_i^0, V_{i+1}^0]| \geq 2q$ for $i = 1, ..., 2l+1$,

iii) $|[V_i^s, V_i^{s+1}]| \geq 2q$ for $s = 0, ..., p_i$ and $i = 1, ..., 2l+1$,

iv) $[V_i^s, V_i^t] = \emptyset$ for $s, t \in \{1, ..., p_i\}$, $|s - t| > 1$ and $(s, t) \neq (0, p_i + 1)$, and $i = 1, ..., 2l+1$,

v) $[V_i^s, V_t^t] = \emptyset$ for $s \in \{1, ..., p_i\}$, $t \in \{1, ..., p_t\}$, $i, t \in \{1, ..., 2l+1\}$, $i \neq t$ (see Figure 2.3).

Let F_i^0 be an edge subset of $[V_0, V_i^{p_i}]$ of q (resp. $q-1$) edges if q is odd (resp. even) and $F = \bigcup_{i=1}^{2l+1} F_i^0$.

With a generalized odd-wheel configuration with q odd (resp. even) we associate the following F-partition inequality induced by the partition π and F,

$$\begin{aligned} x(\delta(\pi) \setminus F) &\geq q\sum_{i=1}^{2l+1} p_i + ql + \frac{q+1}{2}, \\ (\text{resp. } x(\delta(\pi) \setminus F) &\geq q\sum_{i=1}^{2l+1} p_i + (q+1)l + \frac{q+2}{2}). \end{aligned} \tag{2.26}$$

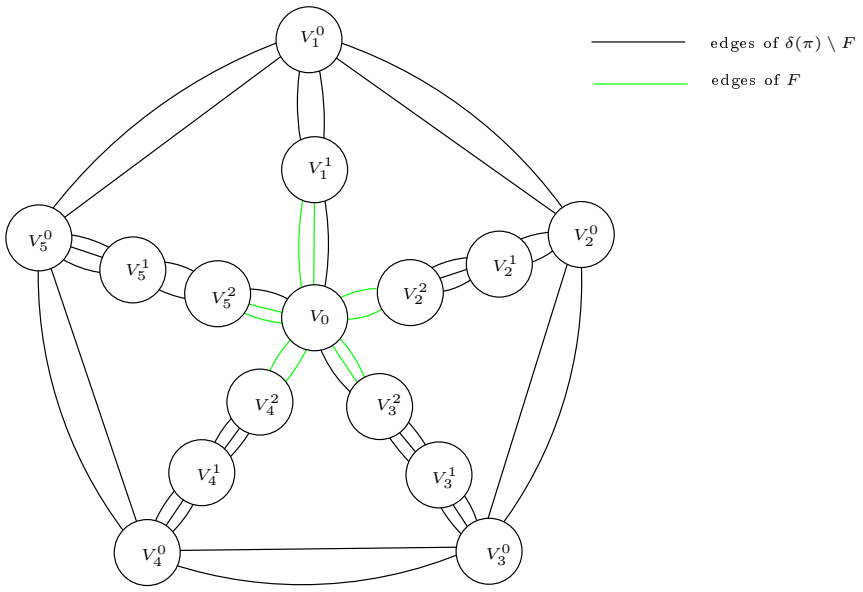


Figure 2.3: A generalized odd-wheel configuration with $k = 4$

Inequalities of type (2.26) will be called *generalized odd-wheel inequalities*. We have the following theorem given without proof, since it follows the same line as that of Theorem 2.2.5

Theorem 2.2.6 *Inequalities (2.26) define facets of kECSP(G).*

2.2.4 SP-partition inequalities

In [21], Chopra introduces a class of valid inequalities for the kECSP when the graph G is outerplanar, k is odd, and each edge can be used more than once. Let $G = (V, E)$ be an outerplanar graph and $k \geq 1$ an odd integer. He showed that if $\pi = (V_1, ..., V_p)$, $p \geq 2$, is a partition of V, then the inequality

$$x(\delta(V_1, ..., V_p)) \geq \left\lceil \frac{k}{2} \right\rceil p - 1, \tag{2.27}$$

is valid for kECSP(G).

Didi Biha and Mahjoub [40] extended this result for general graphs and when each edge can be used at most once. They showed that if G is a graph and $\pi = (V_1, ..., V_p)$, $p \geq 2$, is a partition of V such that G_π is series-parallel and k is odd, then inequality

(2.27) is valid for kECSP(G). They called inequalities (2.27) *SP-partition inequalities* (SP stands for series-parallel). They also described necessary conditions for inequality (2.27) to be facet defining, and showed that if G is series-parallel and k is odd, then kECSP(G) is defined by the trivial, cut and SP-partition inequalities. Further conditions for inequalities (2.27) to be facet defining are given in the following theorems. But before, we give the next two lemmas which describe structural properties of the solutions of the kECSP which satisfy inequalities (2.27) with equality. Note that, in the following results, the indices are taken modulo p.

Lemma 2.2.5 [40] *Let $\overline{x} \in P(G,k)$ and $\pi = (V_1, ..., V_p)$, $p \geq 2$, a partition of V which induces a series-parallel graph. If the SP-partition inequality induced by π is tight for $\overline{x}$, then*

$$\overline{x}([V_i, V_j]) \leq \left\lceil \frac{k}{2} \right\rceil, \text{ for all } i, j \in \{1, ..., p\},\ i \neq j. \tag{2.28}$$

Moreover, if (2.28) is tight for x for a given i and j with $i < j$, then the partition π' obtained by contracting V_i and V_j is also tight for x.

Lemma 2.2.6 *Let $\overline{x}$ be an integer solution of $P(G,k)$ and $\pi = (V_1, ..., V_p)$, $p \geq 2$, be a partition of V such that G_π is series-parallel. Let also $t \in \{1, ..., p\}$, such that the set V_t is adjacent to exactly two elements of π, say V_{t-1} and V_{t+1}. Then $\overline{x}$ satisifies at least one of these inequalities*

$$x([V_t, V_{j_0}]) \geq \left\lceil \frac{k}{2} \right\rceil \text{ with } j_0 \in \{t-1, t+1\}. \tag{2.29}$$

Moreover, if $\overline{x}$ satisifies with equality the inequality (2.27) induced by π, then

$$\overline{x}([V_t, V_{j_0}]) = \left\lceil \frac{k}{2} \right\rceil.$$

Proof. Let $\overline{x} \in \mathbb{R}^E$ be an integer solution of $P(G,k)$. Suppose, w.l.o.g., that $\overline{x}([V_t, V_{t-1}]) \geq \overline{x}([V_t, V_{t+1}])$ and that $j_0 = t-1$. As $\overline{x} \in P(G,k)$, we have that

$$\overline{x}(\delta(V_t)) = \overline{x}([V_t, V_{t-1}]) + \overline{x}([V_t, V_{t+1}]) \geq k.$$

As $\overline{x}$ is integer, this yields $\overline{x}([V_t, V_{t-1}]) \geq \lceil \frac{k}{2} \rceil$.

Now if $\overline{x}$ satisfies with equality the SP-partition inequality induced by π, then, by Lemma 2.2.5, $\overline{x}([V_t, V_{t-1}]) \leq \lceil \frac{k}{2} \rceil$, implying, together with the previous result, that

$$\overline{x}([V_t, V_{t-1}]) = \left\lceil \frac{k}{2} \right\rceil.$$

□

Theorem 2.2.7 *Let $G = (V, E)$ be a $(k+1)$-edge connected graph and $k \geq 3$ an odd integer. Let $\pi = (V_1, ..., V_p)$, $p \geq 2$, be a partition of V such that G_π is series-parallel. The SP-partition inequality induced by π defines a facet of kECSP(G), different from the trivial inequalities, only if*

i) G_π is 2-node-connected,

ii) G_π is outerplanar,

iii) $|[V_i, V_{i+1}]| \geq \lceil \frac{k}{2} \rceil$ for $i = 1, ..., p$.

Proof.

i) First observe that G_π is k-node-connected with $1 \leq k \leq 2$. In fact, since G_π is series-parallel, it contains a node which is adjacent to exactly two other nodes. This implies that the node-connectivity of G_π is at most 2. Moreover, as G is connected, G_π is also connected. Thus $k \geq 1$. We will show in the following that in fact $k = 2$. Suppose, on the contrary, that $k = 1$, that is G_π is 1-node-connected. Thus there exists a node $v_{i_0} \in V_\pi$ and two node sets W_1 and W_2 of V_π such that $(\{v_{i_0}\}, W_1, W_2)$ forms a partition of V_π and $[W_1, W_2] = \emptyset$ (see Figure 2.4).

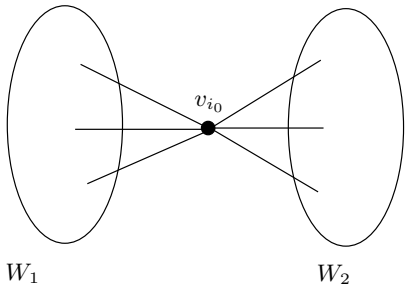


Figure 2.4: A 1-node-connected graph

Let $p_i = |W_i|$, $i = 1, 2$, and π_1 (resp. π_2) be the partition obtained by contracting the sets of π which correspond to the nodes of W_2 (resp. W_1) toghether with those corresponding to v_{i_0}. Clearly, G_{π_i}, $i = 1, 2$, is series-parallel. Thus, the following inequalities are valid for kECSP(G)

$$x(\delta(\pi_i)) \geq \left\lceil \frac{k}{2} \right\rceil (p_i + 1) - 1, \text{ for } i = 1, 2. \tag{2.30}$$

As $[W_1, W_2] = \emptyset$, by summing the inequalities (2.30), we get

$$x(\delta(\pi)) \geq \left\lceil \frac{k}{2} \right\rceil (p_1 + p_2 + 2) - 2 = \left\lceil \frac{k}{2} \right\rceil p - 1 + \left\lceil \frac{k}{2} \right\rceil - 1. \tag{2.31}$$

As $k \geq 3$, we have that $\left\lceil \frac{k}{2} \right\rceil - 1 > 0$, implying that the inequality (2.27) induced by π is dominated by those induced by π_1 and π_2, and hence, cannot define a facet.

ii) Suppose that G_π is series-parallel but not outerplanar, that is one cannot draw G_π in the plane as a cycle with non crossing chords. Thus, there exist two consecutive sets of π, say V_i and V_{i+1}, such that there exist two sets, W_i^1, W_i^2, of elements of π satisfying the following conditions (see Figure 2.5)

a) $[W_i^1, W_i^2] = \emptyset$,

b) $[W_i^j, V_i] \neq \emptyset \neq [W_i^j, V_{i+1}]$ for $j = 1, 2$.

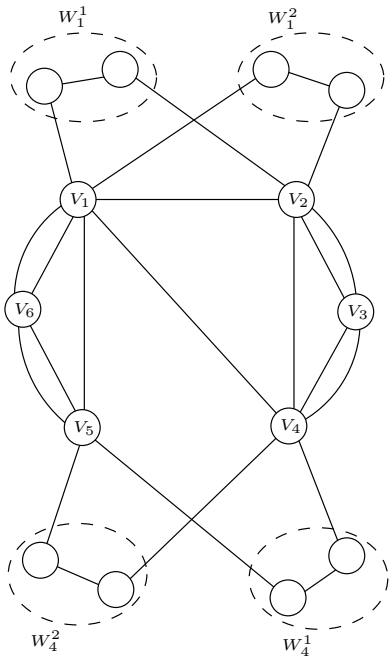


Figure 2.5: A partition inducing a series-parallel but not outerplanar graph

Let $I = \{i \in \{1, ..., p\} \mid V_i, V_{i+1} \in \pi$ and there exist $W_i^1,\ W_i^2 \subseteq V_\pi$ satisfying Conditions a) and b)$\}$. Hence, $I \neq \emptyset$. Let π' be the partition obtained by contracting

together the sets V_i, V_{i+1}, W_i^1, W_i^2, for every $i \in I$. Clearly, $G_{\pi'}$ is outerplanar. Let p_i^1 (resp. p_i^2) be the number of elements of π that are included in W_i^1 (resp. W_i^2), and $p_i = p_i^1 + p_i^2$. Also let $r = \sum_{i \in I} p_i$ and $\pi_{W_i^j}$, $i \in I$, $j \in \{1, 2\}$, be the partition obtained from π by contracting together every set of π which is not in W_i^j (see Figure 2.6).

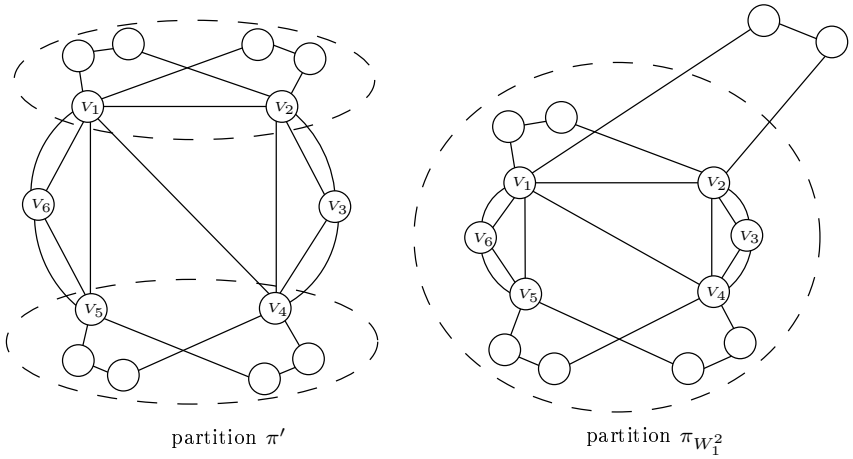


Figure 2.6: Two partitions π' and $\pi_{W_i^j}$

Obviously, the graph $G_{\pi_{W_i^j}}$ is series-parallel. Thus, the following inequalities are valid for kECSP(G),

$$x(\delta(\pi')) \geq \left\lceil \frac{k}{2} \right\rceil (p - r - |I|) - 1 \text{ (inequality (2.27) induced by } \pi'), \tag{2.32}$$

$$x(\delta(\pi_{W_i^1})) \geq \left\lceil \frac{k}{2} \right\rceil (p_i^1 + 1) - 1, \text{ for all } i \in I \text{ (inequality (2.27) induced by } \pi_{W_i^1}), \tag{2.33}$$

$$x(\delta(\pi_{W_i^2})) \geq \left\lceil \frac{k}{2} \right\rceil (p_i^2 + 1) - 1, \text{ for all } i \in I \text{ (inequality (2.27) induced by } \pi_{W_i^2}, \tag{2.34}$$

$$x([V_i, V_{i+1}]) \geq 0 \text{ (trivial inequalities).} \tag{2.35}$$

By summing these inequalities, we get

$$x(\delta(\pi)) \geq \left\lceil \frac{k}{2} \right\rceil p - 1 + |I|(\left\lceil \frac{k}{2} \right\rceil - 2). \tag{2.36}$$

If $k = 3$, the right hand side of (2.36) is the same as that of (2.27) induced by π. Therefore inequality (2.27) is redundant with respect to (2.32), (2.33), (2.34) and (2.35), and hence cannot define a facet.

If $k \geq 4$, since $|I| \geq 1$, the right hand side of (2.36) is greater than that of (2.27). Therefore, (2.27) is dominated by (2.32), (2.33), (2.34) and (2.35), and hence cannot define a facet.

iii) Let $ax \geq \alpha$ denotes the SP-partition inequality induced by π and suppose that this inequality defines a facet of kECSP(G) different from the trivial inequalities. Suppose that there exists an integer $i \in \{1, ..., p\}$ such that $|[V_i, V_{i+1}]| \leq \frac{k-1}{2}$. Let e_i be a fixed edge of $[V_i, V_{i+1}]$. As $ax \geq \alpha$ is different from inequality $x(e_i) \leq 1$, there exists a solution $\overline{x} \in k$ECSP(G) such that $a\overline{x} = \alpha$ and $\overline{x}(e_i) = 0$. We distinguish two cases.

Case 1. The set V_i or V_{i+1} is exactly adjacent to two elements of π. W.l.o.g. we will suppose that V_i is adjacent to V_{i-1} and V_{i+1} only. As $|[V_i, V_{i+1}]| \leq \frac{k-1}{2}$ and $\overline{x}(e_i) = 0$, we have $\overline{x}([V_i, V_{i+1}]) \leq \frac{k-1}{2} - 1$ and $\overline{x}([V_{i-1}, V_i]) \geq \frac{k+1}{2} + 1$, which contradicts Lemma 2.2.5.

Case 2. The sets V_i and V_{i+1} are both adjacent to at least three elements of π (see Figure 2.7).

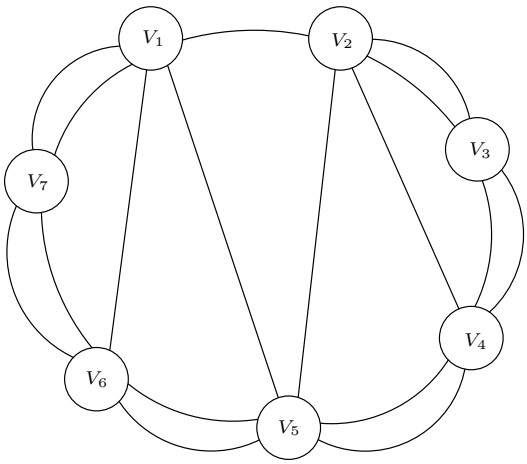


Figure 2.7: The sets V_1 and V_2 are both adjacent to at least three elements of π

Observe that, as G_π is outerplanar and hence series-parallel, one can obtain from π a two-size partition by applying repeatidly the following operation. Let $\pi^j = (V_1^j, ..., V_{p_j}^j)$ be a SP-partition of G and an element $V_{i_0}^j$ incident to exactly two elements $V_{i_0-1}^j$ and $V_{i_0+1}^j$ of π_j. By Lemma 2.2.6, we have either $\overline{x}([V_{i_0}^j, V_{i_0-1}^j]) = \frac{k+1}{2}$ or $\overline{x}([V_{i_0}^j, V_{i_0+1}^j]) = \frac{k+1}{2}$. W.l.o.g., we will suppose that $\overline{x}([V_{i_0}^j, V_{i_0-1}^j]) = \frac{k+1}{2}$ since $i_0 - 1$ and $i_0 + 1$ play the same role. Then, the operation consists in contracting the sets $V_{i_0-1}^j$ and $V_{i_0}^j$ and

considering the partition $\pi^{j+1} = (V_1^{j+1}, ..., V_{p_{j+1}}^{j+1})$ where

$$\begin{aligned} V_i^{j+1} &= V_i^j && \text{for } i = 1, ..., i_0 - 2, \\ V_{i_0-1}^{j+1} &= V_{i_0-1}^j \cup V_{i_0}^j, \\ V_i^{j+1} &= V_{i+1}^j && \text{for } i = i_0, ..., p_j - 1. \end{aligned}$$

We will say that $V_{i_0}^j$ is *merged* with $V_{i_0-1}^j$. Note that each partition π^j induces an outerplanar subgraph of G and that we apply $p-2$ times the operation to obtain a two-size partition from π. Also note that, by Lemma 2.2.5, the SP-partition inequality induced by each partition π^j is tight for $\overline{x}$.

Let π^{j_0} be the first partition obtained by the application of this procedure and such that there exists a node set $V_r^{j_0}$ of π^{j_0} which is adjacent to exactly two elements, say $V_{r-1}^{j_0}$ and $V_{r+1}^{j_0}$, and such that either $V_i \subseteq V_r^{j_0}$ or $V_{i+1} \subseteq V_r^{j_0}$. W.l.o.g., we will suppose that $V_i \subseteq V_r^{j_0}$ and $V_{i+1} \subseteq V_{r+1}^{j_0}$. Remark that π^{j_0} is obtained by the application of the procedure to π^{j_0-1} and $V_s^{j_0-1}$, for some $s \in \{1, ..., p_{j_0-1}\}$, with $V_s^{j_0-1}$ adjacent to exactly two elements of π^{j_0-1}.

Since π^{j_0} is the first partition that we have meet during the successive applications of the procedure and which satisfies the above condition, the partition $\pi^{j_0-1} = (V_1^{j_0-1}, ..., V_{p_{j_0-1}}^{j_0-1})$ is necessarily such that

1. $V_s^{j_0-1}$ is adjacent to exactly two elements $V_{s-1}^{j_0-1}$ and $V_{s+1}^{j_0-1}$,
2. $V_i \subseteq V_{s-1}^{j_0-1}$ and $V_{i+1} \subseteq V_{s+2}^{j_0-1}$,
3. $V_{s-1}^{j_0-1}$ is adjacent to exactly three elements and $V_{s+2}^{j_0-1}$ is adjacent to at least three elements.

One can suppose, w.l.o.g., that $V_s^{j_0-1}$ has been merged with $V_{s-1}^{j_0-1}$ to obtain π^{j_0} (see Figure 2.8).

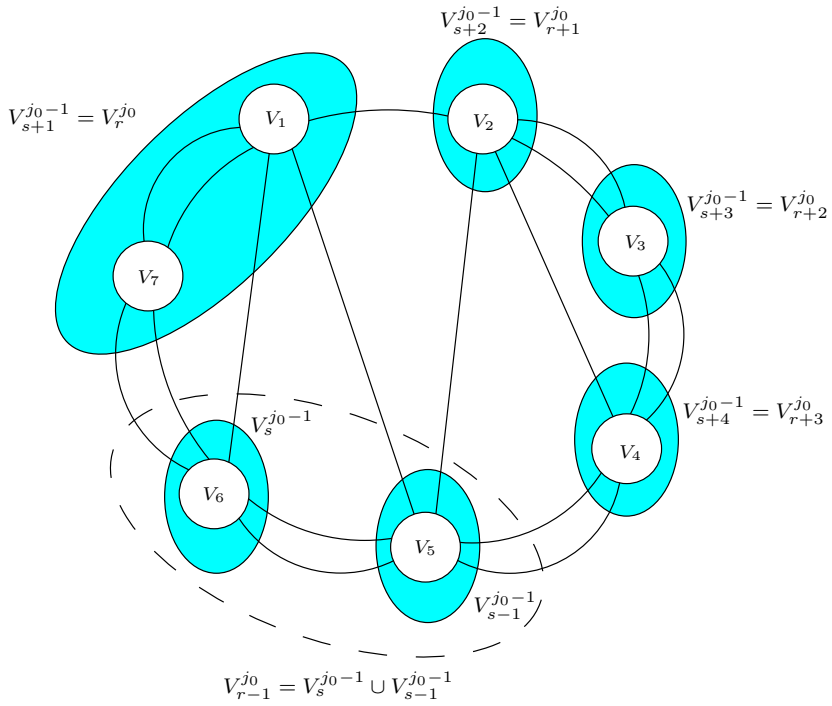


Figure 2.8: Partitions π^{j_0-1} and π^{j_0}.

Now, since by assumption $V_i \subseteq V_r^{j_0}$ and $V_{i+1} \subseteq V_{r+1}^{j_0}$, we have that $|[V_r^{j_0}, V_{r+1}^{j_0}]| \geq |[V_i, V_{i+1}]|$. We are going to show that in fact $|[V_r^{j_0}, V_{r+1}^{j_0}]| = |[V_i, V_{i+1}]|$. Suppose the contrary, that is to say that there exists an edge $e \in [V_r^{j_0}, V_{r+1}^{j_0}] \setminus [V_i, V_{i+1}]$. Clearly, there exist two elements V_t and $V_{t'}$ of π such that $e \in [V_t, V_{t'}]$ and $V_t \subseteq V_r^{j_0}$ and $V_{t'} \subseteq V_{r+1}^{j_0}$. Since G_π is outerplanar, and hence its nodes can be drawn on a cycle with no crossing chords, and since V_i and V_{i+1} are consecutive on this cycle, the node set V_t comes before V_i and $V_{t'}$ comes after V_{i+1} on this cycle (see Figure 2.9 for an illustration).

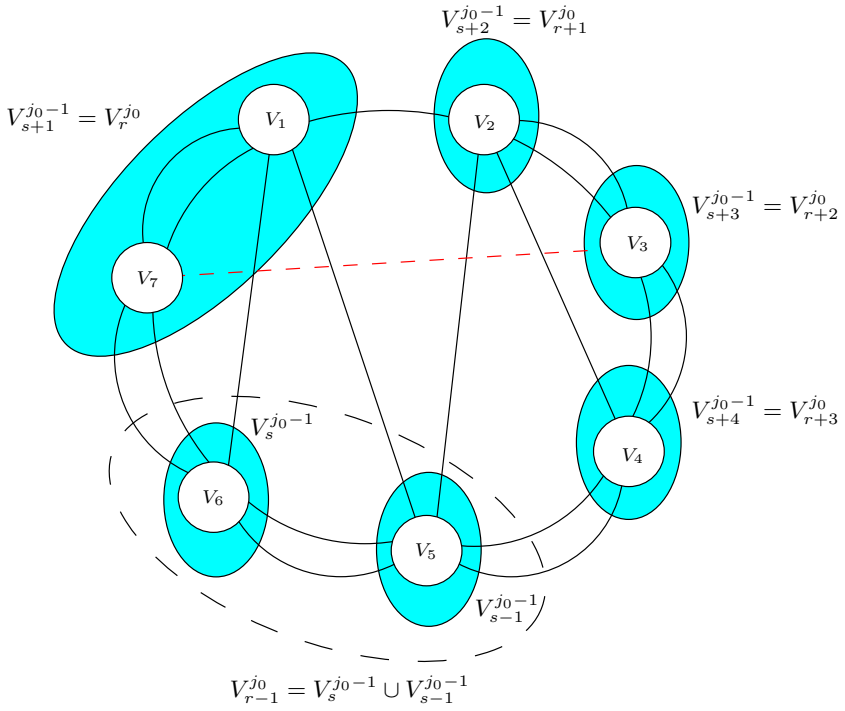


Figure 2.9: An edge of $e \in [V_r^{j_0}, V_{r+1}^{j_0}] \setminus [V_1, V_2]$. Here $e \in [V_t, V_{t'}]$ with $t = 7$ and $t' = 3$.

However, in this situation, any edge $e \in [V_t, V_{t'}]$ is a chord which necessarily crosses the edges of $\delta(V_i \cup V_{i+1})$ (see Figure 2.9), contradicting the fact that G_π is outerplanar. Thus $|[V_r^{j_0}, V_{r+1}^{j_0}]| = |[V_i, V_{i+1}]|$. Therefore, as $|[V_i, V_{i+1}]| \leq \frac{k-1}{2}$ and $\overline{x}(e_i) = 0$, we have that $\overline{x}([V_r^{j_0}, V_{r+1}^{j_0}]) \leq \frac{k-1}{2} - 1$ and $\overline{x}([V_r^{j_0}, V_{r-1}^{j_0}]) \geq \frac{k+1}{2} + 1$, which contradicts Lemma 2.2.5 and ends the proof. □

The following theorem gives some sufficient conditions for inequalities (2.27) to be facet defining.

Theorem 2.2.8 *Let $G = (V, E)$ be a graph and $k \geq 3$ an odd integer. Let $\pi = (V_1, ..., V_p)$, $p \geq 2$, be a partition of V such that G_π is outerplanar and 2-node-connected. Then the SP-partition inequality induced by π is facet defining for kECSP(G), if the following conditions hold*

i) *$G[V_i]$ is $(k+1)$-edge connected for $i = 1, ..., p$,*

ii) *$|[V_i, V_{i+1}]| \geq \lceil \frac{k}{2} \rceil$, $i = 1, ..., p$*
(see Figure 2.10 for an illustration with $k = 3$).

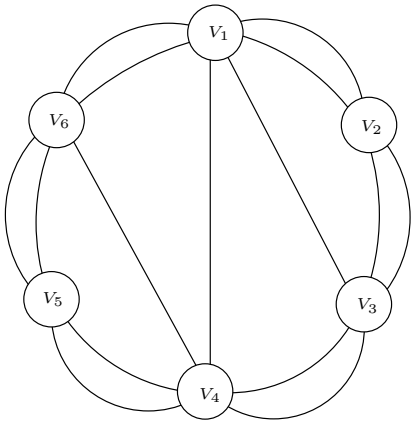


Figure 2.10: An outerplanar configuration with $k = 3$

Proof. Note that since G_π is outerplanar and Conditions 1) and 2) hold, G is $(k+1)$-edge connected. It then follows that kECSP(G) is full dimensional. Let us denote by $ax \geq \alpha$ the SP-partition inequality induced by π and let $\mathcal{F} = \{x \in k\text{ECSP}(G) \mid ax = \alpha\}$. Clearly, $\mathcal{F}$ is a proper face of kECSP(G). Now suppose that there exists a facet defining inequality $bx \geq \alpha$ different from the trivial inequalities such that $\mathcal{F} \subseteq \{x \in k\text{ECSP}(G) \mid bx = \alpha\}$. We will show as before that $b = a$.

Let T_i be an edge subset of $[V_i, V_{i+1}]$, $i = 1, ..., p$, of $\frac{k+1}{2}$ edges and let $T_i' = T_i \setminus \{g_i\}$, where g_i is a fixed edge of T_i. Consider

$$E_0 = \bigcup_{i=1}^{p} E(V_i),$$

$$E_1 = (\bigcup_{i=1}^{p} T_i) \setminus \{g_{i_0}\} \text{ for some } i_0 \in \{1, ..., p\},$$

$$E_2 = E_1 \cup E_0.$$

Note that $g_{i_0} \notin E_2$ and $g_{i_0+1} \in E_2$. Since by Condition 1) the subgraphs induced by the node sets $V_1, ..., V_p$ are $(k+1)$-edge connected, it is not hard to see that E_2 and $E_2' = (E_2 \setminus \{g_{i_0+1}\}) \cup \{g_{i_0}\}$ induce k-edge-connected subgraphs of G. Since x^{E_2} and $x^{E_2'}$ belong to $\mathcal{F}$, we have that $bx^{E_2} = bx^{E_2'} = \alpha$ and hence $b(g_{i_0}) = b(g_{i_0+1})$. As g_{i_0} and g_{i_0+1} are arbitrary edges of T_{i_0} and T_{i_0+1}, respectively, it follows that $b(e) = b(e')$ for all $e \in T_{i_0}$ and $e' \in T_{i_0+1}$. Moreover, since T_{i_0} and T_{i_0+1} are arbitrary subsets of $[V_{i_0}, V_{i_0+1}]$ and $[V_{i_0+1}, V_{i_0+2}]$, respectively, we obtain that $b(e) = b(e')$ for all $e \in [V_{i_0}, V_{i_0+1}]$ and

$e' \in [V_{i_0+1}, V_{i_0+2}]$, $i_0 = 1, ..., p$. Consequently, by symmetry, we get

$$b(e) = b(e') \text{ for all } e, e' \in \bigcup_{i=1}^{p} [V_i, V_{i+1}]. \tag{2.37}$$

Now let $e \in [V_{i_0}, V_{j_0}]$, $i_0, j_0 \in \{1, ..., p\}$ with $|i_0 - j_0| > 1$. Note that $T_0 = T_p$, $T_{-1} = T_{p-1}$ and $T'_0 = T'_p$. Consider the edge sets

$$E_4 = (E_2 \setminus \{g_{i_0-1}\}) \cup \{e\},$$
$$E'_4 = (E_4 \setminus \{e\}) \cup \{g_{i_0}\}.$$

Using Lemma 2.2.2 and the fact that E_2 induces a k-edge-connected subgraph of G, we can see that E_4 and E'_4 induce k-edge-connected subgraphs of G. Since x^{E_4} and $x^{E'_4}$ belong to $\mathcal{F}$, it follows that $bx^4 = bx^{E'_4} = \alpha$, and hence $b(e) = b(g_{i_0})$. By (2.37) this yields

$$b(e) = b(e') \text{ for all } e, e' \in \delta(\pi).$$

Since $ax^{E_2} = bx^{E_2} = \alpha$, we obtain that $b(e) = 1$ for all $e \in \delta(\pi)$.

Next, we will show that $b(e) = 0$ for all $e \in E_0$. Consider the edge set

$$E_5 = E_2 \setminus \{e\} \text{ for some } e \in E_0.$$

Since $G[V_i]$, $i = 1, ..., p$, are $(k+1)$-edge connected, E_5 induces a k-edge-connected subgraph of G. As x^{E_2} and x^{E_5} belong to $\mathcal{F}$, we have that $bx^{E_2} = bx^{E_5} = \alpha$, and thus $b(e) = 0$ for all $e \in E_0$.

In consequence we get $b = a$ and the proof is complete. □

Chopra [21] described a lifting procedure for inequalities (2.27) which can be presented as follows. Let $G = (V, E)$ be a graph and $k \geq 3$ an odd integer. Let $G' = (V, E \cup L)$ be a graph obtained from G by adding an edge set L. Let $\pi = (V_1, ..., V_p)$ be a partition of V such that G_π is series-parallel. Then the following inequality is valid for kECSP(G')

$$x(\delta_G(V_1, ..., V_p)) + \sum_{e \in L \cap \delta_{G'}(V_1, ..., V_p)} a(e)x(e) \geq \left\lceil \frac{k}{2} \right\rceil p - 1, \tag{2.38}$$

where $a(e)$ is the length (in terms of edges) of a shortest path in G_π between the endnodes of e, for all $e \in L \cap \delta_{G'}(V_1, ..., V_p)$.

We will call inequalities of type (2.38) *lifted SP-partition inequalities*. Chopra [21] also showed that, when G is outerplanar, inequality (2.38) defines a facet of kECSP(G') if G is maximal outerplanar, that is to say G is outerplanar and if we add a new edge in G the new graph is not outerplanar. In the following we show that under the same conditions, an inequality of type (2.38) also defines a facet of kECSP(G).

Before this, we give the following lemma whose proof can be found in [21].

Lemma 2.2.7 [21] *Let $G = (V, E)$ be a maximal outerplanar graph which is 2-node connected. Let u, v be two nodes of G and P_1 and P_2 two node-disjoint paths between u and v. Also let $U = \{u_0, ..., u_{r_1}\}$, $r_1 \geq 2$ and $W = \{w_0, ..., w_{r_2}\}$, $r_2 \geq 2$, the node sets of P_1 and P_2 respectively, with $u_0 = w_0 = u$ and $u_{r_1} = u_{r_2} = v$. Remark that $U \cap W = \{u, v\}$ and $V = U \cup W$. If $l \geq 2$ is the length of a shortest path between u and v in G, then there exists at least $l - 1$ egdes $e = u_i w_i$ such that $u_i \in U \setminus \{u, v\}$ and $w_i \in W \setminus \{u, v\}$.*

Theorem 2.2.9 *Let $G = (V, E)$ be a graph and $\pi = (V_1, ..., V_p)$, $p \geq 2$, be a partition of V such that $G_\pi = (V_\pi, E_\pi)$ is outerplanar. Let $\overline{G} = (V, \overline{E})$ be a graph such that $\overline{E} = E \cup \{e_1, ..., e_l\}$, $l \geq 1$. The lifted SP-partition inequality induced by π on $\overline{G}$ defines a facet of kECSP$(\overline{G})$ if the following conditions holds.*

1. *G_π is 2-node-connected and maximal outerplanar,*
2. *$|[V_i, V_{i+1}]| \geq \lceil \frac{k}{2} \rceil$, i = 1,...,p, (modulo p),*
3. *$G[V_i]$ is $(k+1)$-edge connected for all $i = 1, ..., p$.*

Proof. Note that if Conditions 1)-3) hold, then G and $\overline{G}$ are both $(k+1)$-edge connected. It then follows that kECSP$(\overline{G})$ is full dimensional.

Let us denote by $ax \geq \alpha$, the lifted SP-partition inequality induced by π on $\overline{G}$ and $\mathcal{F} = \{x \in k\text{ECSP}(\overline{G}) \mid ax = \alpha\}$. By Conditions 1)-3), the restriction of $ax \geq \alpha$ to G defines a facet of kECSP(G). Thus, $\mathcal{F} \neq \emptyset$ and is a proper face of kECSP$(\overline{G})$. Now suppose that there exists a facet defining inequality $bx \geq \alpha$ different from the trivial inequalities such that $\mathcal{F} \subseteq \{x \in k\text{ECSP}(\overline{G}) \mid bx = \alpha\}$. We will show that $b = a$.

Let $V_\pi = \{v_1, ..., v_p\}$, where v_i corresponds to the set V_i, $i = 1, ..., p$, and let $\overline{G}_\pi = (V_\pi, \overline{E}_\pi)$ be the subgraph of $\overline{G}$ induced by π. Note that $E_\pi \subseteq \overline{E}_\pi$. Since Conditions 1)-3) hold, by Theorem 2.2.8, the SP-partition inequality induced by π on G defines a facet of kECSP(G). Using a proof similar to that of Theorem 2.2.8, one can show that $b(e) = 0$, for all $e \in (\bigcup_{i=1}^{p} \overline{E}(V_i))$, and $b(e) = 1$, for all $e \in E_\pi$. In the following, we are going to show that $b(e) = a(e)$ for all $e \in \{e_1, ..., e_l\}$. Recall that for all $e \in \overline{E}_\pi \setminus E_\pi$, $a(e)$ is the length of a shortest path in G_π between the endnodes of e.

Let T_i be an edge subset of $[V_i, V_{i+1}]$, $i = 1, ..., p$, of $\frac{k+1}{2}$ edges and $T_i' = T_i \setminus \{g_i\}$, where g_i is a fixed edge of T_i. Let $\overline{e} = uv \in \{e_1, ..., e_l\}$ and P_1 and P_2 be two paths in G_π between u and v. Also let r be the length of a shortest path between u and v in G_π. Let U and W denote the node sets of P_1 and P_2 respectively. By Lemma 2.2.7, there exist $r-1$ edges $f_i \in E_\pi$, $i \in \{1, ..., r-1\}$, whose endnodes are in U and W, respectively. We let $w_{i_0} = u$ and $w_{i_0}, ..., w_{i_0+r-1}$ be the endnodes of the edges f_i, $i = 1, ..., r-1$, in W.

Let

$$\overline{E}_1 = \{f_1, ..., f_{r-1}\} \cup (\bigcup_{j=0}^{r-1} T'_{i_0+j}) \cup (\bigcup_{i=1}^{i_0-1} T_i) \cup (\bigcup_{i=i_0+r}^{p} T_i) \cup (\bigcup_{i=1}^{p})\overline{E}(V_i).$$

Obviously, $\overline{E}_1$ induces a solution of the kECSP on $\overline{G}$ and its incidence vector, $x^{\overline{E}_1}$, satisfies $ax \geq \alpha$ with equality. Let $g_i \in T_i$, for $i \in \{1, ..., p\} \setminus \{i_0, ..., i_0 + r - 1\}$, and consider the edge set

$$\overline{E}_2 = (\overline{E}_1 \cup \{\overline{e}\}) \setminus \{g_i,\ i = i_0 - r, ..., i_0 - 1\}.$$

It is not hard to see that $\overline{E}_2$ induces a solution of the kECSP on $\overline{G}$. Moreover, $x^{\overline{E}_2}$ satisfies $ax \geq \alpha$ with equality. This implies that $bx^{\overline{E}_1} = bx^{\overline{E}_2} = \beta$. Thus,

$$bx^{\overline{E}_2} = bx^{\overline{E}_1} + b(e) - \sum_{i=i_0-r}^{i_0-1} b(g_i).$$

Since $g_i \in E_\pi$, $i = i_0 - r, ...i_0 - 1$, and hence $b(g_i) = 1$, we have that $b(\overline{e}) = r$. Therefore, for an edge $e \in \{e_1, ..., e_l\}$, $b(e) = a(e)$.

From this, we get $b(e) = a(e)$, for all $e \in \overline{E}$ and hence, we have $b = a$, which ends the proof of the theorem. □

2.2.5 Partition Inequalities

In this section we present a further class of inequalities, valid for kECSP(G), introduced by Grötschel et al. in [66], that generalizes the cut inequalities. These inequalities, called *partition inequalities*, are defined as follows.

Let $\pi = (V_1, ..., V_p)$, $p \geq 3$, be a partition of V. The *partition inequality* induced by π is given by

$$x(\delta(V_1, ..., V_p)) \geq \left\lceil \frac{kp}{2} \right\rceil. \tag{2.39}$$

If kp is even, then inequality (2.39) is redundant with respect to the cut inequalities. Grötschel et al. [66] gave sufficient conditions for the partition inequalities (2.39) to be facet defining.

Note that the partition inequalities are not a special case of the F-partition inequalities. In fact, if we consider a partition $\pi = (V_0, V_1, ..., V_p)$, $p \geq 2$, the partition inequality induced by π is

$$x(\delta(V_0, V_1, ..., V_p)) \geq \left\lceil \frac{k(p+1)}{2} \right\rceil. \tag{2.40}$$

However the F-partition inequality induced by π and $F = \emptyset$ is given by

$$x(\delta(V_0, V_1, ..., V_p)) \geq \left\lceil \frac{kp}{2} \right\rceil. \tag{2.41}$$

One can remark that inequality (2.40) dominates inequality (2.41).

2.3 Reduction operations

In this section, we are going to describe some graph reduction operations which will be utile for our Branch-and-Cut algorithm. These operations are based on the concept of critical extreme points of $P(G, k)$ introduced by Fonlupt and Mahjoub [49] for $k = 2$ and extended by Didi Biha and Mahjoub [39] for $k \geq 3$.

2.3.1 Description

Before describing these operations, we shall first introduce some notation and definition. Let $G = (V, E)$ be a graph and $k \geq 2$ an integer. If $\overline{x}$ is a solution of $P(G, k)$, we will

denote by $E_0(\overline{x})$, $E_1(\overline{x})$ and $E_f(\overline{x})$ the sets of edges $e \in E$ such that $\overline{x}(e) = 0$, $\overline{x}(e) = 1$ and $0 < \overline{x}(e) < 1$, respectively. We also denote by $C_d(\overline{x})$ the set of degree tight cuts $\delta(u)$ such that $\delta(u) \cap E_f(\overline{x}) \neq \emptyset$, and by $C_p(\overline{x})$ the set of proper tight cuts $\delta(W)$ with $\delta(W) \cap E_f(\overline{x}) \neq \emptyset$. Let $\overline{x}$ be an extreme point of $P(G,k)$. Thus there is a set of cuts $C_p^*(\overline{x}) \subseteq C_p(\overline{x})$ such that $\overline{x}$ is the unique solution of the system

$$S(\overline{x}) \begin{cases} x(e) = 0 & \text{for all } e \in E_0(\overline{x}); \\ x(e) = 1 & \text{for all } e \in E_1(\overline{x}); \\ x(\delta(u)) = k & \text{for all } \delta(u) \in C_d(\overline{x}); \\ x(\delta(W)) = k & \text{for all } \delta(W) \in C_p^*(\overline{x}). \end{cases}$$

Note that the system $S(\overline{x})$ cannot contain an equation $x(\delta(W)) = k$ such that $\delta(W) \cap E_f(\overline{x}) = \emptyset$. Such an equation is redundant with respect to $x(e) = 0$, $e \in E_0(\overline{x})$, and $x(e) = 1$, $e \in E_1(\overline{x})$.

Suppose that $\overline{x}$ is fractional. Let $\overline{x}'$ be a solution obtained by replacing some (but at least one) fractional components of $\overline{x}$ by 0 or 1 (and keeping all the other components of $\overline{x}$ unchanged). If $\overline{x}'$ is a point of $P(G,k)$, then it can be written as a convex combination of extreme points of $P(G,k)$. If $\overline{y}$ is such an extreme point, then $\overline{y}$ is said to be *dominated* by $\overline{x}$, and we write $\overline{x} \succ \overline{y}$. Note that if $\overline{x}$ dominates $\overline{y}$, then $\{e \in E \mid 0 < \overline{y}(e) < 1\} \subset \{e \in E \mid 0 < \overline{x}(e) < 1\}$, $\{e \in E \mid \overline{x}(e) = 0\} \subseteq \{e \in E \mid \overline{y}(e) = 0\}$ and $\{e \in E \mid \overline{x}(e) = 1\} \subseteq \{e \in E \mid \overline{y}(e) = 1\}$. The relation $\succ$ defines a partial ordering on the extreme points of $P(G,k)$. The minimal elements of this ordering (*i.e.*, the extreme points x for which there is no extreme point y such that $x \succ y$) correspond to the integer extreme points of $P(G,k)$. The minimal extreme points of $P(G,k)$ are called extreme points of *rank* 0. An extreme point x is said to be of *rank* p, if x only dominates extreme points of rank $\leq p-1$ and if it dominates at least one extreme point of rank $p-1$. We notice that if $\overline{x}$ is an extreme point of rank 1 and if we replace one fractional component of $\overline{x}$ by 1, keeping unchanged the other integral components, we obtain a feasible solution $\overline{x}'$ of $P(G,k)$ which can be written as a convex combination of integer extreme points of $P(G,k)$.

Didi Biha and Mahjoub [39] introduced the following reduction operations with respect to a solution $\overline{x}$ of $P(G,k)$.

θ_1: delete an edge $e \in E$ such that $\overline{x}(e) = 0$;

θ_2: contract a node subset $W \subseteq V$ such that $G[W]$ is k-edge-connected and $\overline{x}(e) = 1$ for all $e \in E(W)$;

θ_3: contract a node subset $W \subseteq V$ such that $|W| \geq 2$, $|\overline{W}| \geq 2$, $|\delta(W)| = k$ and $E(\overline{W})$ contains at least one edge with fractional value;

θ_4: contract a node subset $W \subseteq V$ such that $|W| \geq 2$, $|\overline{W}| \geq 2$, $G[W]$ is $\lceil \frac{k}{2} \rceil$-edge connected, $|\delta(W)| = k+1$ and $\overline{x}(e) = 1$ for all $e \in E(W)$.

Starting from a graph G and a solution $\overline{x} \in P(G,k)$ and applying θ_1, θ_2, θ_3, θ_4, we obtain a reduced graph G' and a solution $\overline{x}' \in P(G',k)$. Didi Biha and Mahjoub [39] showed that $\overline{x}'$ is an extreme point of $P(G',k)$ if and only if $\overline{x}$ is an extreme point of $P(G,k)$. Moreover, they showed the following results.

Lemma 2.3.1 [39] *$\overline{x}'$ is an extreme point of rank 1 of $P(G',k)$ if and only if $\overline{x}$ is an extreme point of rank 1 of $P(G,k)$.*

Lemma 2.3.2 [39] *If $C_p^*(\overline{x}) = \emptyset$, then the graph induced by $E_f(\overline{x})$ is an odd cycle $C \subseteq E$ such that*

i) $\overline{x}(e) = \frac{1}{2}$ *for all* $e \in C$,

ii) $\overline{x}(\delta(u)) = k$ *for all* $u \in V(C)$.

An extreme point $\overline{x}$ of $P(G,k)$ will be said *critical* if it is of rank 1 and none of the operations θ_1, θ_2, θ_3, θ_4 can be applied to it. If such an extreme point satisfies the assumption of Lemma 2.3.2, then it violates the following F-partition inequality

$$\sum_{e \in C} x(e) \geq \frac{|C|+1}{2}.$$

Hence the critical extreme points of $P(G,k)$ that satisfy the assumption of Lemma 2.3.2 can be separated in polynomial time.

We will use operations θ_1, θ_2, θ_3, θ_4 in our Branch-and-Cut algorithm for the kECSP. As we will see, we use them as a preprocessing for the separation procedures.

2.3.2 Reduction operations and valid inequalities

Given a fractional solution $\overline{x}$ of $P(G,k)$, we let $G' = (V', E')$ and $\overline{x}'$ be obtained by repeated applications of operations θ_1, θ_2, θ_3, θ_4 with respect to $\overline{x}$.

As pointed out above, $\overline{x}'$ is an extreme point of $P(G',k)$ if and only if $\overline{x}$ is an extreme point of $P(G,k)$. Moreover, we have the following lemmas which can be easily seen.

Lemma 2.3.3 *Let $a'x \geq \alpha$ be an F-partition inequality (resp. partition inequality) valid for kECSP(G') induced by a partition $\pi' = (V'_0, V'_1, ..., V'_p)$, $p \geq 2$, (resp. $\pi' = (V'_1, ..., V'_p)$, $p \geq 3$) of V'. Let $\pi = (V_0, V_1, ..., V_p)$, $p \geq 2$, (resp. $\pi = (V_1, ..., V_p)$, $p \geq 3$) be the partition of V obtained by expanding the subsets V'_i of π'. Let $ax \geq \alpha$ be an inequality such that*

$$a(e) = \begin{cases} a'(e) & \text{for all } e \in E', \\ 1 & \text{for all } e \in (E \setminus E') \cap \delta_G(\pi), \\ 0 & \text{otherwise.} \end{cases}$$

Then $ax \geq \alpha$ is valid for kECSP(G). Moreover, if $a'x \geq \alpha$ is violated by $\overline{x}'$, then $ax \geq \alpha$ is violated by $\overline{x}$.

Lemma 2.3.4 *Let $a'x \geq \alpha$ be an odd path inequality (resp. SP-partition inequality) valid for kECSP(G') induced by a partition $\pi' = (W'_1, W'_2, V'_1, ..., V'_{2p})$, $p \geq 2$ (resp. $\pi = (V'_1, ..., V'_p)$, $p \geq 3$). Let $\pi = (W_1, W_2, V_1, ..., V_{2p})$, $p \geq 2$ (resp. $\pi = (V_1, ..., V_p)$, $p \geq 3$), be the partition of V obtained by expanding the elements of π'. Let $ax \geq \alpha$ be the corresponding lifted odd path inequality (resp. lifted SP-partition inequality) obtained from $a'x \geq \alpha$ by application of the lifting procedure described in Section 2.2.2 (resp. Section 2.2.4) for the edges of $E \setminus E'$. Then $ax \geq \alpha$ is violated by $\overline{x}$, if $a'x \geq \alpha$ is violated by $\overline{x}'$.*

Lemmas 2.3.3 and 2.3.4 show that looking for an odd path, F-partition, SP-partition or a partition inequality violated by $\overline{x}$ reduces to looking for such inequality violated by $\overline{x}'$ on G'. Note that this procedure can be applied for any solution of $P(G, k)$ and may, in consequence, permit to separate fractional solutions which are not necessarily extreme points of $P(G, k)$. In consequence, for more efficiency, our separation procedures will be performed on the reduced graph G'. The violated inequalities generated in G' with respect to $\overline{x}'$ are lifted to violated inequalities in G with respect to $\overline{x}$ using Lemmas 2.3.3 and 2.3.4.

Chapter 3

Branch-and-Cut algorithm for the kECSP

In this chapter, we describe a Branch-and-Cut algorithm for the kECSP. Our aim is to address the algorithmic applications of the theoritical results presented in the previous sections and describe some strategic choices made in order to solve that problem. So, let us assume that we are given a graph $G = (V, E)$ and a weight vector $w \in \mathbb{R}^E$ associated with the edges of G. Let $k \geq 3$ be the connectivity requirement for each node of V.

3.1 Branch-and-Cut algorithm

3.1.1 Description

We describe the framework of our algorithm. To start the optimization we consider the following linear program given by the degree cuts associated with the vertices of the graph G together with the trivial inequalities, that is

$$\begin{aligned} &\text{Min} \sum_{e \in E} w(e)x(e) \\ &x(\delta(u)) \geq k \qquad \text{for all } u \in V, \\ &0 \leq x(e) \leq 1 \qquad \text{for all } e \in E. \end{aligned}$$

The optimal solution $\overline{y} \in \mathbb{R}^E$ of this relaxation of the kECSP is feasible for the problem if $\overline{y}$ is an integer vector that satisfies all the cut inequalities. Usually, the solution $\overline{y}$ is

not feasible for the kECSP, and thus in each iteration of the Branch-and-Cut algorithm, it is necessary to generate further inequalities that are valid for the kECSP but violated by the current solution $\overline{y}$. For this, one has to solve the so-called *separation problem*. This consists, given a class of inequalities, in deciding whether the current solution $\overline{y}$ statisfies all the inequalities of this class, and if not, in finding an inequality that is violated by $\overline{y}$. An algorithm solving this problem is called a *separation algorithm*. The Branch-and-Cut algorithm uses the inequalities previously described and their separations are performed in the following order

1. cut inequalities,
2. SP-partition inequalities,
3. odd path inequalities,
4. F-partition inequalities,
5. partition inequalities.

We remark that all inequalities are global (*i.e.*, valid for all the Branch-and-Cut tree) and several inequalities may be added at each iteration. Moreover, we go to the next class of inequalities only if we haven't found any violated inequalities in the current class. Our strategy is to try to detect violated inequalities at each node of the Branch-and-Cut tree in order to obtain the best possible lower bound and thus limit the number of generated nodes. Generated inequalities are added by sets of 200 or fewer at a time.

Now we describe the separation procedures used in our Branch-and-Cut algorithm. These are all heuristic procedures except that for the cut inequalities which is performed using an exact polynomial-time algorithm. The procedures are applied on G' with weights $(\overline{y}'(e), e \in E')$ associated with its edges where $\overline{y}'$ is the restriction on E' of the current LP-solution $\overline{y}$ (G' and $\overline{y}'$ are obtained by repeated applications of operations θ_1, θ_2, θ_3, θ_4).

3.1.2 Separation of cut inequalities

The separation of the cut inequalities (2.3) can be performed by computing minimum cuts in G'. This can be done in polynomial time using Gusfield algorithm [68]. This algorithm produces the so-called *Gomory-Hu tree* with the property that for all pairs

of nodes s, $t \in V'$, the minimum (s,t)-cut in the tree is also a minimum (s,t)-cut in the graph G'. The algorithm requires $|V'|-1$ maximum flow computations. The maximum flow computations are handled by the efficient Goldberg and Tarjan algorithm [58] that runs in $O(m'n' \log \frac{n'^2}{m'})$ time where m' and n' are the number of edges and nodes of G', respectively. Thus our separation algorithm for the cut inequalities is exact and runs in $O(m'n'^2 \log \frac{n'^2}{m'})$ time.

3.1.3 Separation of odd path inequalities

In what follows, we consider the separation of the odd path inequalities (2.4). For this, we need the following lemma.

Lemma 3.1.1 *Let $x \in \mathbb{R}^E$ be a fractional solution of $P(G,k)$ and $\pi = (W_1, W_2, V_1, ..., V_{2p})$, $p \geq 2$, a partition of V, which induces an odd path configuration. If each edge set $[V_i, V_{i+1}]$, $i = 1, ..., 2p-1$, contains an edge with fractional value and*

$$x([V_{i-1}, V_i]) + x([V_i, V_{i+1}]) \leq 1 \text{ for } i = 2, ..., 2p-1,$$

then the odd path inequality induced by π is violated by x.

Proof. As $x([V_{i-1}, V_i]) + x([V_i, V_{i+1}]) \leq 1$, $i = 2, ..., 2p-1$, we have that

$$x([V_{2s-1}, V_{2s}]) + x([V_{2s}, V_{2s+1}]) \leq 1 \text{ for } s = 1, ..., p-1, \tag{3.1}$$
$$x([V_{2s}, V_{2s+1}]) + x([V_{2s+1}, V_{2s+2}]) \leq 1 \text{ for } s = 1, ..., p-1. \tag{3.2}$$

By multiplying inequality (3.1) by $\frac{p-s}{p}$ and inequality (3.2) by $\frac{s}{p}$ and summing the resulting inequalities, we obtain

$$\sum_{i \in I} x([V_i, V_{i+1}]) + \sum_{i \in \overline{I}} \frac{p-1}{p} x([V_i, V_{i+1}]) \leq p-1, \tag{3.3}$$

where $I = \{2, 4, 6, ..., 2p-2\}$ and $\overline{I} = \{1, 2, ..., 2p-1\} \setminus I$. Because each set $[V_i, V_{i+1}]$, $i = 1, ..., 2p-1$, contains an edge with fractional value, we have that $x([V_i, V_{i+1}]) < 1$ for all $i \in \overline{I}$. Hence

$$\sum_{i \in \overline{I}} x([V_i, V_{i+1}]) < p. \tag{3.4}$$

By multiplying inequality (3.4) by $\frac{1}{p}$ and summing the resulting inequality and inequality (3.3), we obtain

$$\sum_{i=1}^{2p-1} x([V_i, V_{i+1}]) < p,$$

and the result follows. □

Our separation heuristic is based on Lemma 3.1.1. The idea is to find a partition $\pi = (W'_1, W'_2, V'_1, ..., V'_{2p})$, $p \geq 2$, which induces an odd path configuration that satisfies the conditions of Lemma 3.1.1. The procedure works as follows. We first look, using a greedy method, for a path $\Gamma = \{e_1, ..., e_{2p-1}\}$, $p \geq 2$, in G' such that the edges $e_1, ..., e_{2p-1}$ have fractional values and $\overline{y}'(e_{i-1}) + \overline{y}'(e_i) \leq 1$, for $i = 2, ..., 2p-1$. If $v'_1, ..., v'_{2p}$ are the nodes of Γ taken in this order when going through Γ, we let $V'_i = \{v'_i\}$, $i = 1, ...2p$, and $T_1 = (\bigcup_{i \in I_1} V'_i) \cup V'_1$ (resp. $T_1 = (\bigcup_{i \in I_1} V'_i) \cup V'_1 \cup V'_{2p}$) if p is odd (resp. even), and $T_2 = (\bigcup_{i \in I_2} V'_i) \cup V'_{2p}$ (resp. $T_2 = (\bigcup_{i \in I_2} V'_i)$) if p is odd (resp. even) where I_1 and I_2 are as defined in Section 2.2.1. In order to determine W'_1 and W'_2, we compute a minimum cut separating T_1 and T_2. If $\delta(W)$ is such a cut with $T_1 \subseteq W$, we let $W'_1 = W \setminus T_1$ and $W'_2 = V' \setminus (W \cup T_2)$. If the partition $\pi = (W'_1, W'_2, V'_1, ..., V'_{2p})$ thus obtained induces an odd path configuration, then, by Lemma 3.1.1, the corresponding odd path inequality is violated by $\overline{y}'$. If not, we apply again that procedure by looking for an other path. In order to avoid the detection of the same path, we label the edges of the detected paths so that they won't appear again when searching for a new path. This procedure is iterated until either a violated odd path inequality is found or all the edges, having fractional values, are labeled. The routine that permits to look for an odd path runs in $O(m'n')$ time. To compute the minimum cut separating T_1 and T_2, we use Goldberg and Tarjan algorithm [58]. Since this algorithm runs in $O(m'n'log\frac{n'^2}{m'})$ time, our procedure is implemented to run in $O(m'^2n' \log \frac{n'^2}{m'})$ time.

In the lifting procedure for inequalities (2.4) given in Section 2.2.2 we have to compute a coefficient λ for some edges $e \in E \setminus E'$. Since the computation of this coefficient is itself a hard problem, and $\lambda \leq 2$, we consider 2 as lifting coefficient for those edges rather than λ.

3.1.4 Separation of F-partition inequalities

Now we discuss our separation procedure for the F-partition inequalities (2.21). These inequalities can be separated in polynomial time using the algorithm of Baïou et al. [6] when k is even and the edge set F is fixed. For the general case, we devised three heuristics to separate them.

Our first heuristic is based on Lemma 2.3.2. As pointed out by that lemma, if $\overline{x}$ is a critical extreme point of $P(G,k)$ such that $C_p^*(\overline{x}) = \emptyset$, then the edges having fractional values with respect to $\overline{x}$ have all a value equal to $\frac{1}{2}$ and form an odd cycle C. Moreover, $\overline{x}(\delta(u)) = k$ for all $u \in V(C)$ and

$$\sum_{e \in C} x(e) \geq \frac{|C|+1}{2},$$

is an F-partition inequality violated by $\overline{x}$. The heuristic works as follows. It starts by determining an odd cycle in G' whose edges have fractional value and nodes are tight. Let $v'_1, ..., v'_p$, $p \geq 3$, be the nodes involved in this cycle. Then we let $V'_i = \{v'_i\}$, for $i = 1, ..., p$, and $V'_0 = V' \setminus \{v'_1, ..., v'_p\}$. We choose the edges of F among those of $\delta(V'_0)$ having values greater than $\frac{1}{2}$ and in such a way that $|F|$ and kp have different parities (if such an edge set F is empty then we look for an other partition). The cycle is obtained by a direct labeling procedure. Hence the heuristic runs in a linear time.

Before introducing our second heuristic, we first give the following lemma.

Lemma 3.1.2 *Let $x \in \mathbb{R}^E$ be a fractional solution of $P(G,k)$ and $\pi = (V_0, V_1, ..., V_p)$, $p \geq 2$, a partition of V such that $x(\delta(V_i)) = k$ for $i = 1, ..., p$. Then an F-partition inequality, induced by π and an edge set $F \subseteq \delta(V_0)$ such that $|F|$ and kp have different parities is violated by x if the following inequality holds*

$$|F| - x(F) + x(\delta(V_0) \setminus F) < 1. \tag{3.5}$$

Proof. As $x(\delta(V_i)) = k$, $i = 1, ..., p$, we have that

$$\sum_{i=1}^{p} x(\delta(V_i)) = 2x(\delta(V_1, ..., V_p)) + x(\delta(V_0)) = kp.$$

This together with (3.5) yield

$$-2x(F) + 2x(\delta(V_0)) + 2x(\delta(V_1, ..., V_p)) < kp - |F| + 1,$$

and thus the statement follows. □

The heuristic is based on Lemma 3.1.2. It starts by determining all the nodes u of V' such that $\overline{y}'(\delta(u)) = k$ and $\delta(u)$ contains at least one edge with fractional value. Let $\{v'_1, ..., v'_p\}$, $p \geq 2$, be the set of such nodes. We consider the partition $(V'_0, V'_1, ..., V'_p)$ such that $V'_i = \{v'_i\}$, for $i = 1, ..., p$, and $V'_0 = V' \setminus \{v_1, ..., v_p\}$, and choose the edges of F in a similar way as in the first heuristic. If inequality (3.5) holds with respect to F and V'_0, then by Lemma 3.1.2 the F-partition inequality corresponding to $(V'_0, V'_1, ..., V'_p)$ and F is violated by $\overline{y}'$.

Before presenting our last heuristic for the F-partition inequalities, let us first remark that a partition $(V'_0, V'_1, ..., V'_p)$ and an edge set $F \subseteq \delta(V'_0)$ may induce a violated F-partition inequality if $\overline{y}'(\delta(V'_0))$ is high and the edges of F are among those of $\delta(V'_0)$ with high values. Our heuristic tries to find such a partition. For this, we first compute a Gomory-Hu tree in G' with the weights $(1 - \overline{y}'(e), e \in E')$ associated with its edges. Then from each proper cut $\delta(W)$ with $V' \setminus W = \{v'_1, ..., v'_p\}$, $p \geq 2$, obtained from the Gomory-Hu tree, we consider the partition $\pi = (V'_0, V'_1, ..., V'_p)$ such that $V'_i = \{v'_i\}$, for $i = 1, ..., p$, and $V'_0 = W$. The edge set F is chosen in a similar way as in the previous heuristics. Since the computation of the Gomory-Hu tree can be done in $O(m'n'^2 \log \frac{n'^2}{m'})$ time, the heuristic runs in $O(m'n'^2 \log \frac{n'^2}{m'})$.

These three heuristics are applied in the Branch-and-Cut algorithm in that order.

3.1.5 Separation of SP-partition inequalities

Now we turn our attention to the separation of the SP-partition inequalities (2.27). These inequalities can be separated in polynomial time using the algorithm of Baöu et al. [6] when G' is series-parallel. That algorithm uses a reduction of the separation problem to the minimization of a submodular function. Recently, Didi Biha et al. [42] devised a pure combinatorial algorithm for the separation of the SP-partition inequalities when the graph is series-parallel. For our purpose, we devised a heuristic to separate inequalities (2.27) in the general case. This heuristic is based on Theorems 2.2.7 and 2.2.8. The main idea of the heuristic is to determine a partition $\pi = (V'_1, ..., V'_p)$, $p \geq 3$, of V' which induces an outerplanar graph such that $|[V'_i, V'_{i+1}]| \geq \lceil \frac{k}{2} \rceil$, $i = 1, ..., p$, (modulo p) (see Figure 2.10), and for every consecutive sets V'_i and V'_j, the edge set $[V'_i, V'_j]$ contains at least one edge with fractional value. To this end, we look in G' for a

path $\Gamma = \{v'_1v'_2, v'_2v'_3, ..., v'_{p-2}v'_{p-1}\}$, $p \geq 3$, such that $|[v'_i, v'_{i+1}]| \geq \lceil \frac{k}{2} \rceil$ and $[v'_i, v'_{i+1}]$ contains one edge or more with fractional value, for $i = 1, ..., p-2$. We then let $V'_i = \{v'_i\}$, $i = 1, ..., p-1$, and $V'_p = V' \setminus \{v'_1, ..., v'_{p-1}\}$. Afterwards, we check by a simple heuristic if the graph G'_π is outerplanar. Finally, we check if the SP-partition inequality induced by π is violated by $\overline{y}'$ or not. If either the graph G'_π is not outerplanar or the SP-partition inequality, induced by π, is not violated by $\overline{y}'$, we apply again this procedure by looking for an other path. In order to avoid the detection of the same path, we label the nodes we met during the search of the previous ones, so that they won't be considered in the search of a new path. This process is iterated until either we find a violated SP-partition inequality or all the nodes of V' are labeled. The heuristic can be implemented to run in $O(m'n')$ time.

3.1.6 Separation of partition inequalities

Now we discuss the separation of the partition inequalities (2.39). First observe that if $\pi = (V'_1, ..., V'_p)$ is a partition of V', with $p \geq 3$ and odd, such that $\overline{y}'(\delta(V'_i)) = k$, for $i = 1, ..., p$, then the partition inequality induced by π is violated by $\overline{y}'$. Thus one can devise a heuristic to separate inequalities (2.39) which consists in finding a partition $\pi = (V'_1, ..., V'_p)$, with $p \geq 3$ and odd, such that $\overline{y}'(\delta(V'_i))$ is as small as possible for $i = 1, ..., p$. To do this, we compute a Gomory-Hu tree, say $\mathcal{T}$, in G' with the weights $(\overline{y}'(e), e \in E')$ associated with its edges. After that, we contract the disjoint node subsets that induce proper tight cuts in $\mathcal{T}$. Let $V'_1, ..., V'_t$ be these sets and $\{v_{t+1}, ..., v_p\} = V' \setminus (\bigcup_{i=1}^{t} V'_i)$. We then consider the partition $(V'_1, ..., V'_t, \{v_{t+1}\}, ..., \{v_p\})$ and check whether or not the corresponding partition inequality is violated by $\overline{y}'$. This algorithm leads to an $O(m'n'^2 \log \frac{n'^2}{m'})$ time complexity.

To store the generated inequalities, we create a pool whose size increases dynamically. All the generated inequalities are put in the pool and are dynamic, *i.e.*, they are removed from the current LP when they are not active. We first separate inequalities from the pool. If all the inequalities in the pool are satisfied by the current LP-solution, we separate the classes of inequalities in the order given above.

3.1.7 Implementation of reduction operations

As mentioned before, the reduction operations θ_1, θ_2, θ_3, θ_4 are applied before the separation procedures. Here we describe the implementation of these reduction operations.

We give only the algorithms for Operations θ_2, θ_3 and θ_4. That of θ_1 is trivial since it consists in deleting every edge $e \in E$ with $\overline{y}(e) = 0$. Note that Operations θ_2, θ_3 and θ_4 are applied on the support graph $G(\overline{y})$.

3.1.7.1 Implementation of Operation θ_2

Operation θ_2 consists in contracting a node set $W \subseteq V$ such that the subgraph $G[W]$ induces a k-edge-connected subgraph and $\overline{y}(e) = 1$ for all $e \in E(W)$.

We apply the following heuristic for Operation θ_2. First, we consider the graph G_1 obtained by deleting from $G(\overline{y})$ all the edges with a fractional value and compute the connected components of G_1. Let $(V_1, ..., V_p)$, $p \geq 1$, be the set of the connected components. Note that G_1 may be connected. Then, we apply the following procedure to every connected component of G_1. Consider a stack Q of node sets, initialized with the sets V_i, $i = 1, ..., p$. Remind that to push a node set W in Q is to put W on the top of Q. Also to pop an element from Q is to remove from Q the node set which is on the top Q. We apply the following algorithm on the sets in Q until Q is empty.

Algorithm 2: Operation θ_2

```
Data: Q = {V_1, ..., V_p}, G(ȳ) = (V, E(ȳ))
Result: Reduced graph G_r = (V_r, E_r)
begin
    while Q is not empty do
        Let W be the top of Q and pop W;
        if |W| ≥ 2 and |V \ W| ≥ 2 then
            if the subgraph induced by W in G(ȳ) does not contain edges with
            fractional value then
                Check if G_1[W] is k-edge-connected or not by computing the
                minimum capacity cut of G_1[W];
                if true then
                    contract W;
                else
                    Let [W_1, W_2] denote the minimum capacity cut of G_1[W];
                    Push W_1 and W_2 on Q;
end
```

To compute the minimum capacity cut of $G_1[W]$, we use Hao and Orlin's algorithm [69] which runs in $O(nm \log \frac{n^2}{m})$ times. Note that given a set V_i, $i = 1, ..., p$, the main

loop of Algorithm 2 contains a number of iterations in $O(\log(|V_i|))$. Each iteration consists at most in checking if the graph induced by W contains edges with fractional value and computating of a minimum capacity cut. Thus, the algorithm for Operation θ_2 runs in $O(\log(n)(nm\log(\frac{n^2}{m}) + m))$. Hence, this procedure is polynomial.

3.1.7.2 Implementation of Operation θ_3

Operation θ_3 consists in contracting a node set W such that $|W| \geq 2$, $|V \setminus W| \geq 2$, $|\delta(W)| = k$ and $E(V \setminus W)$ contains edges with fractional values. We devise the following heuristic for this operation. First we give 1 as capacity for every edge of $G(\overline{y})$ and compute a Gomory-Hu tree on it. Let T be the tree obtained. Observe that every edge of T with weight k induces a cut $\delta(W)$ of exactly k edges in $G(\overline{y})$. We apply the procedure described below on every k-capacity cut $\delta(W)$ obtained from T until we find a candidate node set to contract or we explore all the k-capacity cuts obtained from T. The procedure is described as follows. If $|W| \geq 2$ and $|V \setminus W| \geq 2$, then we check if the subgraph induced by $V \setminus W$ in $G(\overline{y})$ contains edges with fractional values or not. If this is the case, then we contract W. If not, then we check if the graph induced by W in $G(\overline{y})$ contains edges with fractional values. If this is the case, then we contract $V \setminus W$ and terminate the procedure.

We repeat this procedure until no contraction is possible by the algorithm.

The implementation for Operation θ_3 is summarized by Algorithm 3.

Algorithm 3: Operation θ_3

```
Data: G(ȳ) = (V, E(ȳ))
Result: Reduced graph G_r = (V_r, E_r)
begin
    repeat
        Give 1 as capacity on the edges of G(ȳ);
        Compute a Gomory-Hu tree T;
        foreach δ(W) obtained from T such that |δ(W)| = k do
            if |W| ≥ 2 and |V \ W| ≥ 2 then
                if G(ȳ)[V \ W] contains edges with fractional values then
                    Contract W;
                    Break;
                else
                    if G(ȳ)[W] contains edges with fractional values then
                        Contract V \ W;
                        Break;
    until no contraction is possible;
end
```

This algorithm contains at most $O(\log(n))$ iterations. Each iteration is composed of the computation of a Gomory-Hu tree and, for every cut $\delta(W)$ obtained in T, the check that $G(\overline{y})[V \setminus W]$ or $G(\overline{y})[W]$ contains edges with fractional values. As the computation of the Gomory-Hu tree runs in $O(mn^2 \log \frac{n^2}{m})$, each iteration runs in $O(mn^2 \log \frac{n^2}{m} + m)$. Thus, the whole algorithm runs in $O(\log(n)(mn^2 \log \frac{n^2}{m} + m))$ and is polynomial.

3.1.7.3 Implementation of operation θ_4

Operation θ_4 consists in contracting a node set W such that $|W| \geq 2$, $|V \setminus W| \geq 2$, $|\delta(W)| = k + 1$, $G[W]$ is $\lceil \frac{k}{2} \rceil$-edge-connected and $\overline{y}(e) = 1$ for all $e \in E(W)$. We propose two heuristics for this operation.

The first heuristic is as follows. We give 1 as capacity for every edge of $G(\overline{y})$ and compute a Gomory-Hu tree on $G(\overline{y})$ with these capacities. If T denotes this tree, one can observe that every edge of T with weight $k + 1$ induces in $G(\overline{y})$ a cut $\delta(W)$ of exactly $k + 1$ edges. For every cut $\delta(W)$ such that $|\delta(W)| = k + 1$ obtained from T, we check if the subgraph $G(\overline{y})[W]$ does not contain any edge with fractional value. If this is the case, then we check if $G(\overline{y})[W]$ is $\lceil \frac{k}{2} \rceil$-edge-connected by computing its

minimum cut. If $G(\overline{y})[W]$ is $\lceil \frac{k}{2} \rceil$-edge-connected, then we contract W. If $G(\overline{y})[W]$ is not $\lceil \frac{k}{2} \rceil$-edge-connected or it contains edges with fractional values, then we perform the same checks on $\overline{W}$. If $G(\overline{y})[\overline{W}]$ does not contain edges with fractional value and is $\lceil \frac{k}{2} \rceil$-edge-connected, then we contract $\overline{W}$. We repeat this algorithm until no contraction is possible.

In the second heuristic, we look for cliques W of $G(\overline{y})$ with $(\lceil \frac{k}{2} \rceil + 1)$ nodes such that $\overline{y}(e) = 1$ for all $E(\overline{y})(W)$ and such that $|\delta(W)| = k + 1$. It is not hard to see that if W is a clique of $(\lceil \frac{k}{2} \rceil + 1)$ nodes, then the subgraph induced by W is $\lceil \frac{k}{2} \rceil$-edge-connected. If such clique exists in $G(\overline{y})$ with $|\delta(W)| = k+1$ and $\overline{y}(e) = 1$ for all $e \in E(\overline{y})(W)$, then we contract W. One can use a greedy algorithm to compute a clique W of $(\lceil \frac{k}{2} \rceil + 1)$ nodes and such that the subgraph induced by W does not contain edges with fractional value. As for the previous heuristic, we repeat this algorithm until no contraction is possible.

These two algorithms are summurized in Algorithms 4 and 5.

Algorithm 4: Operation $\theta_4 - 1$

```
Data: G(ȳ) = (V, E(ȳ))
Result: Reduced graph G_r = (V_r, E_r)
begin
    repeat
        Give 1 as capacity on the edges of G(ȳ);
        Compute a Gomory-Hu tree T;
        foreach δ(W) obtained from T such that |δ(W)| = k + 1 do
            if |W| ≥ 2 and |V \ W| ≥ 2 then
                if G(ȳ)[W] does not contain edges with fractional value then
                    Compute the minimum cut of G(ȳ)[W];
                    if G(ȳ)[W] is ⌈k/2⌉-edge-connected then
                        Contract W;
                        Break;
    until no contraction is possible;
end
```

Algorithm 5: Operation $\theta_4 - 2$

Data: $G(\overline{y}) = (V, E(\overline{y}))$
Result: Reduced graph $G_r = (V_r, E_r)$
begin
  repeat
    Search a clique W of $G(\overline{y})$ on $(\lceil \frac{k}{2} \rceil + 1)$ nodes and such that $\overline{y}(e) = 1$ for all $e \in E(\overline{y})(W)$;
    if W *exists and* $|W| \geq 2$ *and* $|V \setminus W| \geq 2$ **then**
      if $|\delta_{G(\overline{y})}(W)| = k + 1$ **then**
        Contract W;
        Break;
  until *no contraction is done;*
end

The minimum cut of a subgraph $G[W]$ is computed using Hao and Orlin's algorithm [69]. As for Operation θ_3, the first heuristic runs in $O(\log(n)(mn^2 \log \frac{n^2}{m} + m))$. It is thus polynomial. For the second algorithm, the greedy algorithm used to find cliques of $G(\overline{y})$ runs in $O(\frac{n^2K^3}{2})$ where $K = max\{|\delta_{G(\overline{y})}(u)|, \text{ for all } u \in V\}$. Remark that in most cases, $|\delta_{G(\overline{y})}(u)| \leq 2k$, for every $u \in V$. We will thus consider that $K \leq 2k$. This implies that the heuristic runs in $O(n^2k^3)$ in most cases, and is polynomial.

Figure 3.1 gives an example of application of Operations θ_3 and θ_4 on a fractional extreme point of $P(G, k)$. The dashed edges have value 0.5 and the plain edges have value 1.

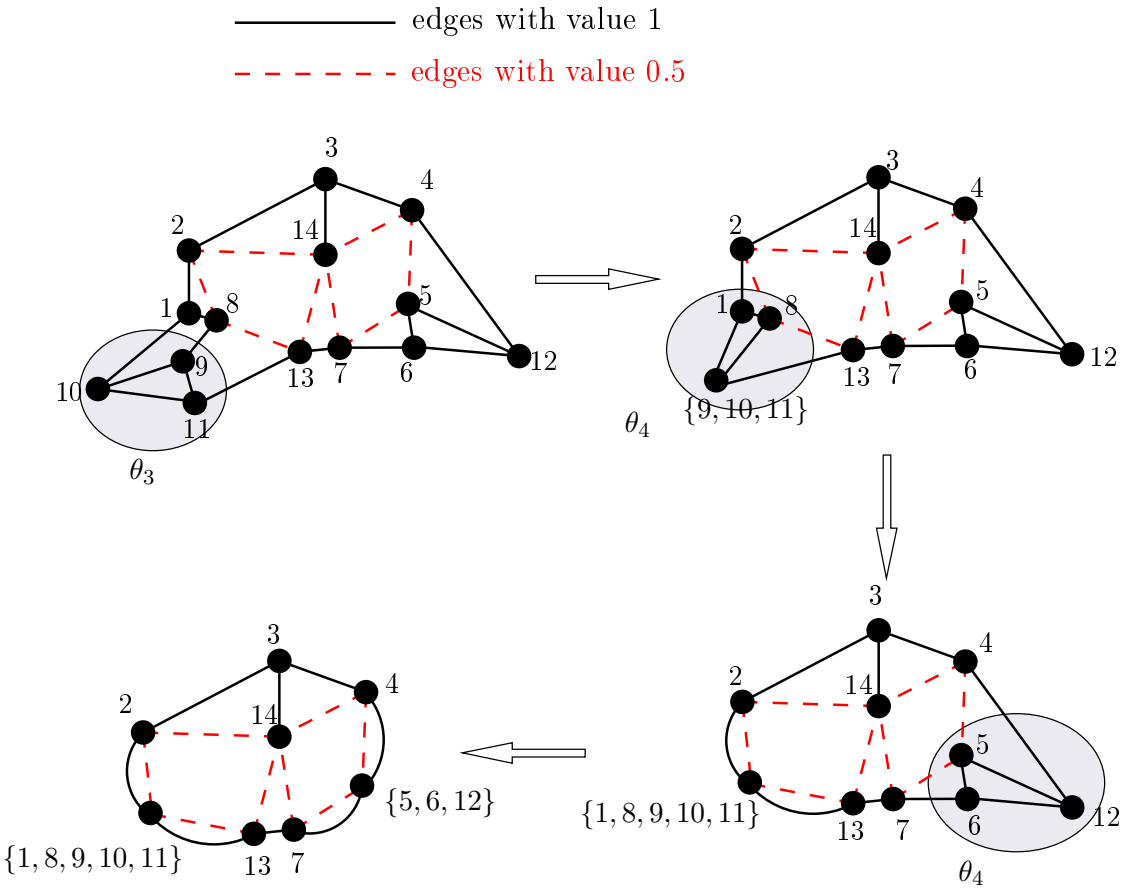


Figure 3.1: Example of application of Operations θ_3 and θ_4 for $k = 3$

On Figure 3.1, we can easily see that the partitions
$\pi_1 = (\{1, 8, 9, 10, 11\}, \{2\}, \{13\}, \{3, 4, 5, 6, 7, 12, 14\})$ and
$\pi_2 = (\{5, 6, 12\}, \{4\}, \{7\}, \{1, 2, 3, 8, 9, 10, 11, 13, 14\})$ induce two SP-partition inequalities that are violated by the underlying fractional solution of the example.

3.1.8 Primal heuristic

Another important issue in the effectiveness of the Branch-and-Cut algorithm is the computation of a good upper bound at each node of the Branch-and-Cut tree. To do this, if the separation procedures do not generate any violated inequality and the current solution $\overline{y}$ is still fractional, then we transform $\overline{y}$ into a feasible solution of the kECSP, say $\hat{\overline{y}}$, by rounding up to 1 all the fractional components of $\overline{y}$. We then try to reduce the weight of the solution thus obtained by removing from the subgraph

$H = (V, \hat{E})$ induced by $\hat{\overline{y}}$ some unecessary edges, that is to say edges which do not affect the k-edge-connectedness of H. To this end, we remove from $\hat{E}$ each edge $e = uv$ such that $|\delta(u) \cap \hat{E}| \geq k+1$ and $|\delta(v) \cap \hat{E}| \geq k+1$. We then check if the resulting edge set, say $\hat{E}'$, induces a k-edge-connected subgraph of G by computing a Gomory-Hu tree. If there exists in $\hat{E}'$ a cut $\delta(W)$, $W \subseteq V$, containing less than k edges, then we add in $\hat{E}'$ edges of $[W, V \setminus W] \setminus \delta(W)$ that have been previously removed from $\hat{E}$ as many as necessary in order to satisfy the cut $\delta(W)$. We do this until the graph $(V, \hat{E}')$ becomes k-edge-connected. Note that we add to each violated cut the edges having the smallest weights.

3.2 Computational results

The Branch-and-Cut algorithm described in the previous section has been implemented in C++, using ABACUS 2.4 alpha [1, 101] to manage the Branch-and-Cut tree, and CPLEX 9.0 [2] as LP-solver. It was tested on a Pentium IV 3.4 Ghz with 1 Gb of RAM, running under Linux. We fixed the maximum CPU time to 5 hours. The test problems were obtained by taking TSP test problems from the TSPLIB library [3]. The test set consists in complete graphs whose edge weights are the rounded euclidian distance between the edge's vertices. The tests were performed for $k = 3, 4, 5$. In all our experiments, we have used the reduction operations described in the previous sections, unless otherwise specified. Each instance is given by its name followed by an extension representing the number of nodes of the graph. The other entries of the various tables are:

NCut	:	number of generated cut inequalities;
NSP	:	number of generated SP-partition inequalities;
NOP	:	number of generated odd path inequalities;
NFP	:	number of generated F-partition inequalities;
NP	:	number of generated partition inequalities;
COpt	:	weight of the optimal solution obtained;
Gap1	:	the relative error between the best upper bound (the optimal solution if the problem has been solved to optimality) and the lower bound obtained at the root node of the Branch-and-Cut tree using only the cut and the trivial inequalities;
Gap2	:	the relative error between the best upper bound (the optimal solution if the problem has been solved to optimality) and the lower bound obtained at the root node of the Branch-and-Cut tree;
NSub	:	number of subproblems in the Branch-and-Cut tree;
TT	:	total CPU time in hours:min:sec.

The instances indicated with "*" are those whose CPU time exceeded 5 hours. For these instances, the gap is indicated in italic.

Our first series of experiments concerns the kECSP for $k = 3$. The instances we have considered have graphs with 14 up to 318 nodes. The results are summarized in Table 3.1. It appears from Table 3.1 that all the instances have been solved to optimality within the time limit except the last five instances. Also we have that four instances (burma14, gr21, fri26, brazil58) have been solved in the cutting plane phase (*i.e.*, no branching is needed). For most of the other instances, the relative error between the lower bound at the root node of the Branch-and-Cut tree and the best upper bound (Gap2) is less than 1%. We also observe that our separation procedures detect a large enough number of SP-partition and F-partition inequalities and seem to be quite efficient.

Our second series of experiments concerns the kECSP with $k = 4, 5$. The results are given in Table 3.2 for $k = 4$ and Table 3.3 for $k = 5$. The instances considered have graphs with 52 up to 561 nodes. Note that for $k = 4$, the SP-partition and partition inequalities are redundant with respect to the cut inequalities (2.3). Thus these inequalities are not considered in the resolution process for $k = 4$, and therefore do not appear in Table 3.2.

Instance	NCut	NSP	NOP	NFP	NP	COpt	Gap1	Gap2	NSub	TT
burma14	4	3	0	0	4	5530	4.67	0.00	1	0:00:01
ulysses16	5	7	1	15	7	11412	1.17	0.39	3	0:00:11
gr21	5	6	1	0	2	4740	1.65	0.00	1	0:00:01
fri26	9	5	0	0	0	1543	1.30	0.00	1	0:00:01
bayg29	14	16	2	33	2	2639	1.76	0.19	7	0:00:01
dantzig42	41	31	6	90	18	1210	2.27	0.68	71	0:00:07
att48	34	34	5	60	9	17499	1.83	0.56	61	0:00:06
berlin52	36	31	12	97	6	12601	1.66	0.45	33	0:00:03
brazil58	46	42	2	36	29	42527	2.67	0.00	1	0:00:05
eil76	9	12	3	298	2	876	0.63	0.06	7	0:00:03
pr76	130	207	72	2231	54	187283	3.9	1.50	6767	0:35:32
rat99	41	26	13	341	23	2029	1.26	0.38	41	0:00:47
kroA100	170	197	31	1207	57	36337	4.64	0.97	4201	0:54:06
kroB100	130	114	37	830	47	37179	2.61	0.73	723	0:08:00
rd100	101	74	11	418	18	13284	1.91	0.43	171	0:03:37
eil101	86	72	21	3604	15	1016	1.06	0.55	1109	0:17:41
lin105	179	198	47	829	68	25530	3.66	0.69	1031	0:22:39
pr107	201	190	34	674	114	70852	2.48	0.84	2071	1:26:49
gr120	50	45	6	588	17	11442	1.12	0.19	99	0:11:15
bier127	46	59	4	276	13	198184	1.50	0.15	11	0:01:55
ch130	121	132	30	1355	40	10400	2.27	0.55	1693	1:05:05
ch150	92	93	19	588	22	11027	2.04	0.41	193	0:20:31
kroA150	155	143	41	845	47	44718	2.27	0.53	1205	1:16:35
kroB150	130	110	16	952	48	43980	2.26	0.31	437	0:38:43
rat195	24	19	3	514	1	3934	0.48	0.06	7	0:08:21
d198	171	105	23	617	59	25624	2.00	0.21	159	1:04:19
gr202	77	69	14	558	22	65729	1.02	0.11	69	0:13:16
*pr226	364	248	35	162	41	-	*11.05*	*9.02*	261	5:00:00
*gr229	179	245	23	1568	94	-	*2.43*	*1.00*	1219	5:00:00
*pr264	275	181	145	668	62	-	*12.56*	*12.29*	69	5:00:00
*a280	142	84	56	2539	59	-	*3.73*	*2.69*	459	5:00:00
*lin318	189	147	15	610	58	-	*6.5*	*4.94*	25	5:00:00

Table 3.1: Results for $k = 3$ with reduction operations.

First observe that for $k = 4$, the CPU time for all the instances is relatively small and most of the instances have been solved in less than 1 minute. We can also observe that 23 instances over 27 are solved in the cutting plane phase. Moreover, a few number of odd path inequalities are generated. However a large enough number of F-partition inequalities is detected. Thus these latter inequalities seem to be very effective for solving the kECSP when k is even. This also shows that the kECSP is easier to solve when k is even, what is also confirmed by the results of Table 3.3 for $k = 5$. In fact, the instance pr264 has been solved for $k = 4$ in 1 second, whereas it could not be solved to optimality for $k = 5$ after 5 hours. The same observation can be done for pr439. Also, we can remark that the CPU time for all the instances when $k = 5$ is higher than that when $k = 4$. For instance, the test problem d198 has been solved in 1h 50mn when $k = 5$, whereas only 16 seconds were needed to solve it for $k = 4$.

Compared to Table 3.1, Tables 3.2 and 3.3 also show that, for the same parity of k, the kECSP becomes easier to solve when k increases. In fact, with $k = 3$, we could not solve to optimality instances with more than 202 nodes, whereas for $k = 5$, we could solve larger instances.

The results for $k = 3, 4, 5$ can also be compared to those obtained by Kerivin et al. [81] for the 2ECSP. It turns out that for the same instances, the problem has been easier to solve for $k = 2$ than for $k = 3$. However, for $k = 4$ the problem appeared to be easier to solve than for $k = 2$. This shows again that the case when k is odd is harder to solve than that when k is even and that the problem becomes easier when k increases with the same parity.

In order to evaluate the impact of the reduction operations θ_1, θ_2, θ_3, θ_4 on the separation procedures, we tried to solve the kECSP, for $k = 3$, without using them. The results are given in Table 3.4.

As it appears from Tables 3.1 and 3.4, the CPU time increased for the majority of the instances when the reduction operations are not used. In particular, for the instance pr107, without the reduction operations, we could not reach the optimal solution after 5 hours, whereas with the reduction operations, it has been solved to optimality after 1h 26mn. Also, the CPU time for the instances ch130 and d198 increased from 1 hour to more than 4 hours. Moreover, we remark that when using the reduction operations, we generate more SP-partition, F-partition and partition inequalities and fewer nodes in the Branch-and-Cut tree. This implies that our separation heuristics are less efficient without the reduction operations. It seems then that the reduction operations play an important role in the resolution of the problem. They permit to strengthen much more the linear relaxation of the problem and accelerate its resolution.

We also tried to measure the effect of the different non-basic classes of inequalities (*i.e.*, inequalities other than cut and trivial inequalities). For this, we have first considered a Branch-and-Cut algorithm for the kECSP with $k = 3$ using only the cut constraints in addition to the trivial ones. As it appears from Table 3.1, for all the instances we have that Gap1 is greater than Gap2. For example, for the instances KroA100 and rat195, the gap is increased by almost 3%.

Furthermore, in this case, we could not solve any of the instances with more than 52 nodes. Even more, after less than 10 minutes of CPU time, the Branch-and-Cut tree got a very big size and the resolution process stops. To illustrate this, take for example the instance brazil58. For this instance, the Branch-and-Cut tree contained 11769 nodes after 10 minutes when the Branch-and-Cut algorithm used only the cut and trivial inequalities, whereas it has been solved without branching when using the other classes of inequalities.

Finally, we tried to evaluate separately the efficiency of each class of the non-basic inequalities. For this, we also considered the case when $k = 3$. We have seen that all the classes of inequalities have a big effect on the resolution of the problem. In particular, the SP-partition inequalities seem to play a central role. This can be seen by considering the instance d198. This instance has been solved in 1h 04mn using all the constraints. However, without the SP-partition inequalities, we could not reach the optimal solution after 5 hours. We also remarked that the gap2 increased when one of these classes of inequalities is not used in the Branch-and-Cut algorithm.

3.3 Concluding remarks

In this chapter, we have studied the k-edge-connected subgraph problem with high connectivity requirement, that is, when $k \geq 3$. We have presented some classes of valid inequalities and described some conditions for these inequalities to be facet defining for the associated polytope. We also discussed separation heuristics for these inequalities. Using these results, we have devised a Branch-and-Cut algorithm for the problem. This algorithm uses some reduction operations.

Our computational results have shown that the odd path, the F-partition, the SP-partition and the partition inequalities are very effective for the problem when k is odd. They have also shown the importance of the F-partition inequalities for the even case. We could also see the importance of our separation heuristics. In particular, our heuristics to separate the SP-partition and F-partition inequalities have appeared

to be very efficient. In addition, the reduction operations have been essential for having a good performance of the Branch-and-Cut algorithm. In fact, they permitted to considerably reduce the size of the graph supporting a fractional solution and to accelerate the separation process.

These experiments also showed that the kECSP is easier to solve when k is even and that, for the same parity of k, the problem becomes easier to solve when k increases.

One of the separation heuristic devised for the F-partition inequalities is based on a partial characterization of the critical extreme points of the linear relaxation of the k-edge-connected subgraph polytope. It would be very interesting to have a complete characterization of these points. This may yield the identification of new facet defining inequalities for the problem. It may also permit to devise more appropriate separation heuristics for the inequalities given in this chapter.

In many real instances, we may consider node-connectivity instead of edge-connectivity. The study presented in this chapter may be very usefull for the k-node-connected subgraph problem for which we require k node-disjoint paths between every pair of nodes.

In addition to the survivability aspect, one can consider the capacity dimensioning of the network. These issues have been mostly treated separately in the literature. It would be interesting to extend the study developed in this chapter to the more general capacitated survivable network design model.

Instance	NCut	NOP	NFP	COpt	Gap2	NSub	TT
berlin52	5	0	2	18295	0.00	1	0:00:01
pr76	3	0	4	266395	0.00	1	0:00:01
kroA100	10	0	11	51221	0.00	1	0:00:47
kroB100	9	5	123	53597	0.08	21	0:00:09
rd100	10	1	91	19130	0.00	1	0:00:05
eil101	0	0	60	1453	0.00	1	0:00:02
lin105	20	1	5	36353	0.00	1	0:00:01
pr107	29	0	0	98381	0.00	1	0:00:01
gr120	6	0	36	16400	0.00	1	0:00:02
bier127	16	0	0	282207	0.00	1	0:00:01
ch130	12	0	132	14854	0.00	1	0:00:05
ch150	12	2	70	15854	0.00	1	0:00:02
kroA150	13	0	27	64249	0.00	1	0:00:02
kroB150	20	0	4	62710	0.00	1	0:00:01
rat195	0	0	37	5750	0.00	1	0:00:13
d198	43	0	71	35404	0.01	3	0:00:16
gr202	13	3	220	94841	0.02	3	0:01:28
pr226	91	0	6	183537	0.00	1	0:00:04
gr229	24	2	15	318565	0.00	1	0:00:03
pr264	59	1	7	122941	0.00	1	0:00:06
a280	3	0	180	6317	0.00	1	0:01:00
pr299	30	0	427	117559	0.00	1	0:00:20
lin318	28	0	2	105000	0.00	1	0:00:06
rd400	21	2	232	36676	0.00	1	0:07:39
pr439	78	3	61	264975	0.02	19	0:02:52
si535	0	0	4	53604	0.00	1	0:00:39
pa561	10	1	306	6724	0.00	1	0:08:37

Table 3.2: Results for $k = 4$.

Instance	NCut	NSP	NOP	NFP	NP	COpt	Gap2	NSub	TT
berlin52	5	2	2	26	2	24845	0.00	1	0:00:01
pr76	2	0	0	52	1	372392	0.00	1	0:00:01
kroA100	5	1	5	76	6	71422	0.04	11	0:00:06
kroB100	6	1	2	83	5	74241	0.01	3	0:00:06
rd100	6	2	6	193	5	26168	0.01	5	0:00:24
eil101	1	0	0	309	0	1938	0.00	1	0:01:10
lin105	9	1	3	119	3	50711	0.00	1	0:00:26
pr107	92	40	57	680	33	132870	0.41	381	0:14:45
gr120	2	0	3	93	3	22024	0.11	27	0:00:17
bier127	22	2	12	450	8	383165	0.09	25	0:04:25
ch130	1	0	0	45	0	20508	0.01	3	0:00:05
ch150	5	0	7	58	1	21791	0.01	37	0:00:50
kroA150	9	0	5	141	3	87950	0.07	11	0:00:19
kroB150	14	1	7	462	6	85583	0.02	11	0:15:39
rat195	1	0	0	508	0	7773	0.00	1	0:20:54
d198	56	9	6	1093	32	47614	0.15	337	1:50:40
gr202	0	0	0	64	0	128990	0.00	1	0:00:31
pr226	142	34	20	661	50	260878	0.58	103	2:38:50
gr229	18	1	11	679	9	434422	0.06	43	0:31:58
*pr264	105	12	38	1327	28	-	*1.78*	43	5:00:00
a280	2	0	2	302	0	8643	0.02	7	0:05:05
pr299	11	3	2	637	1	161576	0.00	1	0:05:12
lin318	24	3	11	1548	11	144341	0.02	7	4:34:39
rd400	11	1	15	691	6	49893	0.01	17	1:29:09
*pr439	46	2	8	746	0	-	*3.46*	1	5:00:00
si535	0	0	0	0	0	79115	0.00	1	0:00:19
pa561	1	0	2	286	1	9161	0.00	1	3:26:58

Table 3.3: Results for $k = 5$.

Instance	NCut	NSP	NOP	NFP	NP	COpt	Gap2	NSub	TT
berlin52	31	28	19	44	4	12601	0.44	15	0:00:04
brazil58	50	27	1	28	31	42527	0.22	3	0:00:07
eil76	9	6	3	102	2	876	0.00	1	0:00:01
pr76	103	168	65	1378	37	187283	1.60	3483	0:38:46
rat99	41	19	10	223	17	2029	0.32	61	0:01:29
kroA100	193	234	47	1765	70	36337	1.42	7575	4:13:38
kroB100	141	142	36	899	38	37179	0.98	1337	0:45:34
rd100	103	84	15	445	21	13284	0.40	233	0:11:40
eil101	77	58	26	2527	12	1016	0.38	801	0:18:50
lin105	161	158	50	569	53	25530	0.61	547	0:34:25
*pr107	218	221	136	1101	104	-	*0.81*	4447	5:00:00
gr120	42	38	6	252	15	11442	0.18	93	0:05:38
bier127	58	56	9	240	12	198184	0.16	17	0:04:43
ch130	141	147	38	1590	45	10400	0.52	2459	4:10:31
ch150	90	76	15	391	23	11027	0.39	107	0:21:07
kroA150	155	135	23	705	56	44718	0.55	1107	3:08:37
kroB150	150	141	22	1006	43	43980	0.31	535	1:55:20
rat195	23	18	7	898	1	3934	0.01	19	0:19:23
d198	192	118	25	720	50	25624	0.27	585	5:03:16
gr202	73	62	13	278	23	65729	0.05	37	0:37:31

Table 3.4: Results for $k = 3$ without reduction operations.

Chapter 4

The k-Edge-Disjoint Hop-Constrained Paths Problem

Given a graph $G = (V, E)$ and two nodes $s, t \in V$, and a positive integer $L \geq 2$, an *L-st-path* in G is a path between s and t of length at most L, where the length is the number of its edges. Given a function $c : E \to \mathbb{R}$ which associates a cost $c(e)$ to each edge $e \in E$ and an integer $k \geq 2$, the *k-Edge-Disjoint Hop-Constrained Paths problem* (kHPP for short) is to find a minimum cost subgraph such that between s and t there exist at least k edge-disjoint L-st-paths.

In this chapter, we consider the kHPP from a polyhedral point of view. In particular, we give a complete description of the associated polytope in the case $L = 3$. We give an integer programming formulation for the problem in this case. In particular, we show that for $L = 3$, the kHPP polytope is given by the so-called st-cut and L-path-cut inequalities together with the trivial inequalities. We also describe necessary and sufficient conditions for these inequalities to be facet defining and show that the kHPP polytope is completely described by the st-cut and L-path-cut toghether with the trivial inequalities. These results generalize those obtained by [75] who give a complete description of the kHPP polytope in the case $k = 2$ and $L = 2, 3$ and by [35] who completely characterize the kHPP polytope when $k \geq 2$ and $L = 2$. This work has led to a technical report submitted for possible publication in Discrete Optimization [13].

The chapter is organized as follows. In next section, we give some preliminary results we will use along this chapter. In Section 4.2, we describe necessary and sufficient conditions for the so-called st-cut and L-path-cut inequalities to be facet defining.

Our main result, which is a complete description of the kHPP polytope for $L = 3$, is presented in Section 4.3. In Section 4.4, we give some concluding remarks.

4.1 Preliminary results

4.1.1 Valid inequalities for the kHPP polytope

Given a graph $G = (V, E)$, two nodes s, t of V and a positive integer $k \geq 2$, we will denote by $k\text{HPP}(G)$ the kHPP polytope that is the convex hull of the incidence vectors of the solutions of the kHPP on G.

If x^F is the incidence vector of the edge set F of a solution of the kHPP, then clearly x^F statisfies the following inequalities:

$$x(\delta(W)) \geq k, \quad \text{for all } st - \text{cut } \delta(W), \tag{4.1}$$

$$0 \leq x(e) \leq 1, \quad \text{for all } e \in E. \tag{4.2}$$

Inequalities (4.1) will be called *st-cut inequalities* and inequalities (4.2) *trivial inequalities*.

In [31], Dahl considers the problem of finding a minimum cost path between two given terminal nodes s and t of length at most L. He describes a class of valid inequalities for the problem and gives a complete characterization of the associated L-path polyhedron when $L \leq 3$. In particular he introduces a class of valid inequalities as follows.

Let V_0, V_1, ..., V_{L+1} be a partition of V such that $s \in V_0$ and $t \in V_{L+1}$, and $V_i \neq \emptyset$ for all $i = 1, ..., L$. Let T be the set of edges $e = uv$, where $u \in V_i$, $v \in V_j$, and $|i - j| > 1$. Then the inequality

$$x(T) \geq 1$$

is valid for the L-path polyhedron.

Using the same partition, this inequality can be generalized in a straightforward way to the kHPP polytope as

$$x(T) \geq k. \tag{4.3}$$

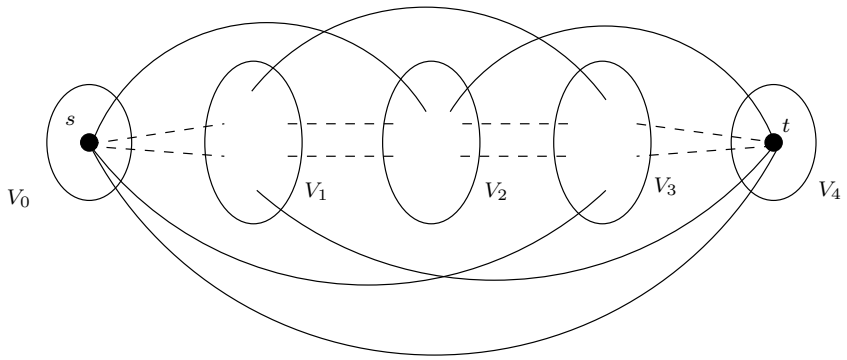


Figure 4.1: Support graph of a 3-path-cut inequality.

The set T is called an L-path-cut, and a constraint of type (4.3) is called an L-path-cut inequality. See Figure 4.1 for an example of a 3-path-cut inequality with $V_0 = \{s\}$ and $V_4 = \{t\}$. Note that T intersects every 3-st-path in at least one edge and each st-cut $\delta(W)$ intersects every st-path. We denote by $P_k(G)$ the polytope given by inequalities (4.1)-(4.3).

4.1.2 Formulation

In this subsection, we give an integer programming formulation for the kHPP. We will show that the st-cut, 3-path-cut and trivial inequalities, together with the integrality constraints suffice to formulate the kHPP as a 0-1 linear program. To this end, we first give a lemma. Its proof can be found in [75].

Lemma 4.1.1 [75] *Let $G = (V, E)$ be an undirected graph and s and t two nodes of V. Suppose that there do not exist k edge-disjoint 3-st-paths in G, with $k \geq 2$. Then there exists a set of at most $k-1$ edges that intersects every 3-st-path.*

Theorem 4.1.1 *Let $G = (V, E)$ be a graph and $k \geq 2$. Then the kHPP is equivalent to the integer program*

$$\text{Min}\left\{cx;\ x \in P_k(G),\ x \in \{0,1\}^E\right\}.$$

Proof. To prove the theorem, it is sufficient to show that every 0-1 solution x of $P_k(G)$ induces a solution of the kHPP. Let us assume the contrary and suppose that x does not induce a solution of the kHPP but satsifies the st-cut and trivial inequalities. We

will show that x necessarily violates at least one 3-path-cut inequality. Let $G(x)$ be the subgraph of G induced by x, that is the graph obtained from G by deleting every edge $e \in E$ such that $x(e) = 0$. As x is not a solution of the problem, $G(x)$ does not contain k edge-disjoint 3-st-paths. By Lemma 4.1.1, it follows that there exist at most $k-1$ edges in $G(x)$ that intersect every 3-st-path. Consider the graph $G'(x)$ obtained from $G(x)$ by deleting these edges. Obviously, $G'(x)$ does not contain any 3-st-path.

We claim that $G'(x)$ contains at least one st-path of length at least 4. In fact, as x is a 0-1 solution and satisfies the st-cut inequalities, $G(x)$ contains at least k edge-disjoint st-paths. Since at most $k-1$ edges were removed from $G(x)$, at least one path remains between s and t. However, since $G'(x)$ does not contain a 3-st-path, that st-path must be of length at least 4.

Now consider the partition $(V_0, ..., V_4)$ of V with $V_0 = \{s\}$, V_i the set of nodes at distance i from s in $G'(x)$ for $i = 1, 2, 3$, and $V_4 = V \setminus (\bigcup_{i=0}^{3} V_i)$, where the distance between two nodes is the length of a shortest path between these nodes. Since there does not exist a 3-st-path in $G'(x)$, it is clear that $t \in V_4$. Moreover, as by the claim above, $G'(x)$ contains an st-path of length at least 4, the sets V_1, V_2 and V_3 are nonempty. Futhermore, no edge of $G'(x)$ is a chord of the partition (that is an edge between two sets V_i an V_j where $|i-j| > 1$). In fact, if there exists an edge $e = v_i v_j \in [V_i, V_j]$ with $|i-j| > 1$ and $i < j$, then v_j is at distance $i+1 < j$, from s, a contradiction.

Thus, the edges deleted from $G'(x)$ are the only edges that may be chords of the partition $G(x)$. In consequence, if T is the set of chords of the partition in G, then $x(T) \leq k-1$. But this implies that the corresponding 3-path-cut inequality is violated by x. □

4.1.3 Disjoint st-paths in directed graphs

Here we will introduce known results related to disjoint st-paths in directed graphs which will be very useful in the following sections.

Given a directed graph $D = (V, A)$, two nodes $s, t \in V$, an integer $k \geq 2$ and a weight function $c(.)$ on the arcs of D, the k *arc-disjoint st-paths problem* (kADPP for short) consists in finding a minimum weight subgraph of D which contains at least k

arc-disjoint paths from s to t. Let kADPP(D) be the convex hull of the solutions of the kADPP on D.

If B is an arc subset of A which induces a solution of the kADPP, then its incidence vector x^B satisfies the following inequalities:

$$x(\delta^+(W)) \geq k, \text{ for all } W \subseteq V, s \in W \text{ and } t \in \overline{W}, \tag{4.4}$$

$$0 \leq x(a) \leq 1, \quad \text{for all } a \in A. \tag{4.5}$$

Conversely, any integral solution of the system given by inequalities (4.4) and (4.5) induces a solution of the kADPP. Inequalities (4.4) are called *st-dicut inequalities* and constraints (4.5) are called *trivial inequalities*. Thus, the kADPP is equivalent to

$$\min\{cx \mid x \text{ satisfies (4.4), (4.5), } x \in \{0,1\}^A\}.$$

Theorem 4.1.2 [96]
The polytope kADPP(D) is full dimensional if and only if every st-dicut $\delta^+(W)$ of D contains at least $k+1$ arcs.

Theorem 4.1.3 *An inequality (4.4), induced by a node set $W \subseteq V$, defines a facet of kADPP(D) if and only if the corresponding st-dicut is minimal inclusionwise and contains at least $k+1$ arcs.*

The following theorem shows that the *st*-dicut and the trivial inequalities suffice to describe the polytope kADPP(G).

Theorem 4.1.4 [96]
The polytope kADPP(G) is completely described by inequalities (4.4) and (4.5).

The following theorem indicates that two node subsets W_1 and W_2 of V that induce tight *st*-dicut inequalities for a solution $y \in k$ADPP(D), can be seen as embedded node sets. This comes from the fact that the sets inducing *st*-dicuts in a graph form a laminar family.

Theorem 4.1.5 [96]
Let W_1 and W_2 be two node subsets of V that induce st-dicuts of D such that $W_1 \cap W_2 \neq \emptyset \neq (V \setminus W_1) \cap W_2$. If the st-dicut inequalities, induced by W_1 and W_2, are tight for a solution x of kADPP(G), then there exists a node set different from W_1 and W_2 contained either in W_1 or in $W_1 \cup W_2$ which induces a tigh st-dicut inequality for x.

These results will be utile in the rest of the chapter for exhibiting some facets of the kHPP polytope, and for proving our main result.

4.2 Facets of kHPP(G)

In this section, we give necessary and sufficient conditions for inequalities (4.1)-(4.3) to define facets. These will be useful in the sequel.

Let $G = (V, E)$ be an undirected graph, s and t two nodes of G and k a positive integer ≥ 2. An edge $e \in E$ is said to be 3-*st-essential* if e belongs to an st-cut or a 3-path-cut of cardinality k. Let E^* be the set of the 3-st-essential edges. We have the following results that can be easily seen to be true.

Theorem 4.2.1 *$dim(kHPP(G)) = |E| - |E^*|$.*

An immediate consequence of Theorem 4.2.1 is the following.

Corollary 4.2.1 *If $G = (V, E)$ is a complete graph such that $|V| \geq k + 2$, then kHPP(G) is full dimensional.*

In the rest of the chapter, we will consider that $G = (V, E)$ is a complete graph with $|V| \geq k + 2$, and which may contain multiple edges. Thus, by Corollary 4.2.1, kHPP(G) is full dimensional.

Lemma 4.2.1 *Let $ax \geq \alpha$ be an inequality which defines a facet of kHPP(G), different from (4.2). Then $a(e) \geq 0$ for all $e \in E$.*

Proof. Let $f \in E$. As $ax \geq \alpha$ is different from facets induced by the trivial inequalities, it is different from $x(f) \leq 1$. Thus, there exists a solution $\overline{x} \in k$HPP(G) such that $a\overline{x} = \alpha$ and $\overline{x}(f) = 0$. Let $\overline{x}'$ be the solution defined by

$$\overline{x}'(e) = \begin{cases} \overline{x}(e), & \text{for all } e \in E \setminus \{f\}, \\ 1 & \text{if } e = f. \end{cases}$$

Clearly, $\overline{x}'$ is a solution of kHPP(G). Hence, $a\overline{x}' = a\overline{x} + a(f) \geq \alpha$, yielding $a(f) \geq 0$. □

The following theorems show when inequalities (4.1)-(4.3) define facets for kHPP(G).

Theorem 4.2.2 *1. Inequality $x(e) \leq 1$ defines a facet of kHPP(G) for all $e \in E$.*

2. Inequality $x(e) \geq 0$ defines a facet of kHPP(G) if and only if either $|V| \geq k+3$ or $|V| = k+2$ and e does not belong neither to an st-cut nor to a 3-path-cut containing exactly $k+1$ edges.

Proof. 1) As $|V| \geq k+2$ and G is complete, the edge set $E_f = E \setminus \{f\}$ is a solution of kHPP, for all $f \in E \setminus \{e\}$. Hence, the sets E and E_f, for all $f \in E \setminus \{e\}$, constitute a set of $|E|$ solutions of the kHPP. Moreover, their incidence vectors satisfy $x(e) = 1$ and are affinelly independant.

2) Suppose that $|V| \geq k+3$. Then G contains $k+2$ node-disjoint st-paths (an edge of $[s,t]$ and $k+1$ paths of the form (s,u,t), $u \in V \setminus \{s,t\}$). Hence any edge set $E \setminus \{f,g\}$, $f,g \in E$, contains k edge-disjoint 3-st-paths among these 3-st-paths.

Consider the $|E|$ edge sets $E \setminus \{e\}$ and $E_f = E \setminus \{e,f\}$ for all $f \in E \setminus \{e\}$. Therefore, these sets induce solutions of the kHPP. Moreover the incidence vectors of these solutions satisfy $x(e) = 0$ and are affinelly independant.

Now suppose that $|V| = k+2$. If e belongs to an st-cut $\delta(W)$ (resp. a 3-path-cut T) with $k+1$ edges, then $x(e) \geq 0$ is redundant with respect to the inequalities

$$\begin{aligned} &x(\delta(W)) \geq k \quad (\text{resp. } x(T) \geq k), \\ &-x(f) \geq -1 \quad \text{for all } f \in \delta(W) \setminus \{e\} (\text{resp. } f \in T \setminus \{e\}), \end{aligned}$$

and cannot hence be facet defining. If e does not belong neither to an st-cut nor to a 3-path-cut with $k+1$ edges, then the edge sets $E \setminus \{e\}$ and E_f, $f \in E \setminus \{e\}$, introduced above, are still solutions of kHPP. Moreover, their incidence vectors satisfy $x(e) = 0$ and are affinelly independant. □

Theorem 4.2.3 *Every st-cut inequality defines a facet of kHPP(G).*

Proof. Let $W \subseteq V$ such that $s \in W$ and $t \in \overline{W}$. Observe that $[s,t] \subseteq \delta(W)$. Let us denote by $ax \geq \alpha$ the st-cut inequality induced by W and let $\mathcal{F}_a = \{x \in k\text{HPP}(G) \mid ax = \alpha\}$. We first show that $\mathcal{F}_a$ is a proper face of kHPP(G). As $|V| \geq k+2$, there exist $W_1 \subseteq W \setminus \{s\}$ and $W_2 \subseteq \overline{W} \setminus \{t\}$ such that $|W_1| + |W_2| = k$. Note that W_1 and W_2

may be empty but not both. Let $F_1 = \{sv,\ v \in W_2\} \cup \{ut,\ u \in W_1\}$ and $E_1 = F_1 \cup E_0$ where $E_0 = E(W) \cup E(\overline{W})$. It is not hard to see that E_1 is a solution of the kHPP whose incidence vector satisfies $ax \geq \alpha$ with equality. Hence, $\mathcal{F}_a \neq \emptyset$ and, therefore, is a proper face of kHPP(G).

Now suppose that there exists a facet defining inequality $bx \geq \beta$ such that $\mathcal{F}_a \subseteq \{x \in k\text{HPP}(G) \mid bx = \beta\}$. We will show that there exists a scalar ρ such that $b = \rho a$.

Consider an edge $e \in F_1$. Clearly, the edge set $E_2 = (E_1 \setminus \{e\}) \cup \{st\}$ is a solution of the kHPP and its incidence vector satisfies $ax \geq \alpha$ with equality. It then follows that $bx^{E_2} = bx^{E_1} - b(e) + b(st)$. Since $x^{E_1} \in \mathcal{F}_a$, we obtain that $b(e) = b(st)$. As e is arbitrary in F_1, this implies that

$$b(e) = b(st) = \rho \text{ for all } e \in F_1. \tag{4.6}$$

Now let $f = uv \in \delta(W) \setminus F_1$, with $u \in W \setminus \{s\}$ and $v \in \overline{W} \setminus \{t\}$. If $u \in W_1$ and $v \in W_2$, then let $E_3 = (E_1 \setminus \{sv, ut\}) \cup \{f, st\}$. Clearly, E_3 is a solution of the kHPP and its incidence vector satisfies $ax \geq \alpha$ with equality. Hence, we have that $bx^{E_3} = bx^{E_1}$. This implies that $b(sv) + b(ut) = b(f) + b(st)$. From (4.6), it follows that $b(f) = \rho$.
If $u \in W_1 \cup \{s\}$ (resp. $u \in W \setminus (W_1 \cup \{s\})$) and $v \in \overline{W} \setminus (W_2 \cup \{t\})$ (resp. $v \in W_2 \cup \{t\}$), by considering the edge set $E_4 = (E_1 \setminus \{ut\}) \cup \{f\}$ (resp. $E_4 = (E_1 \setminus \{sv\}) \cup \{f\}$), we similarly obtain that $b(f) = \rho$.
If $u \notin W_1$ and $v \notin W_2$, then one can consider the solution $E_5 = (E_1 \setminus \{e\}) \cup \{f\}$, where e is an edge of F_1, and obtain along the same lines that $b(f) = \rho$.

Thus, toghether with (4.6), this yields

$$b(e) = \rho \text{ for all } e \in \delta(W).$$

Now let $e \in E_0$, and suppose, w.l.o.g., that $e \in E(W)$. If e does not belong to a 3-st-path of E_1, then the edge set $E_6 = E_1 \setminus \{e\}$ also induces a solution of the kHPP and satisfies $ax \geq \alpha$ with equality. We then have that $bx^{E_6} = bx^{E_1}$ implying $b(e) = 0$.

If e belongs to a 3-st-path of E_1, say (su, ut), then the edge set $E_7 = (E_1 \setminus \{su, ut\}) \cup \{st\}$ induces a solution of the kHPP and its incidence vector satisfies $ax \geq \alpha$ with equality. It then follows that $bx^{E_7} = bx^{E_1}$ and hence $b(st) = b(su) + b(ut)$. As by (4.6), $b(ut) = b(st)$, we get $b(e) = 0$.

Consequently, we have that

$$b(e) = \begin{cases} \rho & \text{for all } e \in \delta(W), \\ 0 & \text{if not.} \end{cases}$$

Thus, $b = \rho a$ with $\rho \in \mathbb{R}$, and the result follows. □

The next theorem describes necessary and sufficient conditions for L-path-cut inequalities to define facets. But before, we give the following lemma.

Lemma 4.2.2 *Let T be an L-path-cut induced by a partition $\pi = (V_0, ..., V_4)$ with $s \in V_0$ and $t \in V_4$. If an edge set $F \subseteq E$ induces a solution of the kHPP such that $x^F(T) = k$, then $F \cap ([s, V_1] \cup [V_3, t] \cup [s, t]) \geq k$. Moreover, if $F \cap [V_1, V_3] \neq \emptyset$, then $F \cap ([s, V_1] \cup [V_3, t] \cup [s, t]) \geq k + 1$.*

Proof. Let $A = [s, V_1] \cup [V_3, t] \cup [s, t]$. Since each 3-$st$-path of F intersects T at least once and $|F \cap T| = k$, F necessarily contains exactly k edge-disjoint 3-st-paths. Moreover, each of these paths intersects T only once. This implies that every 3-st-path of F is of the form

i) (su_1, u_1u_2, u_2t), (su_2, u_2u_3, u_3t), (su_1, u_1t), (su_3, u_3t), (st) or

ii) (su_1, u_1u_3, u_3t).

If P is one of these st-paths, then $|P \cap A| = 1$ (resp. $|P \cap A| = 2$) if P is of type i) (resp. ii)). Thus, $|F \cap A| \geq k$.

Now if $F \cap [V_1, V_3] \neq \emptyset$, then F contains at least one path of type ii) and therefore $|F \cap A| \geq k + 1$. □

Theorem 4.2.4 *An inequality (4.3), induced by a partition $\pi = (V_0, ..., V_4)$ with $s \in V_0$ and $t \in V_4$, defines a facet of kHPP(G), different from a trivial inequality, if and only if*

1. $|V_0| = |V_4| = 1$;
2. $|[s, V_1]| + |[V_3, t]| + |[s, t]| \geq k + 1$.

Proof. Let T be the 3-path-cut induced by π. Let $ax \geq \alpha$ denote the 3-path-cut inequality produced by T and $\mathcal{F} = \{x \in k\text{HPP}(G) \mid ax = \alpha\}$.

Necessity.
1) We will show that if $|V_0| \geq 2$, inequality $x(T) \geq k$ does not define a facet. The case where $|V_4| \geq 2$ follows by symmetry. Suppose that $|V_0| \geq 2$ and consider the partition $\pi' = (V'_0, ..., V'_4)$ given by

$$\begin{aligned} V'_0 &= \{s\}, \\ V'_1 &= V_1 \cup (V_0 \setminus \{s\}), \\ V'_i &= V_i, \ i = 2, 3, 4. \end{aligned}$$

The partition π' produces a 3-path-cut inequality $x(T') \geq k$, where $T' = T \setminus [V_0 \setminus \{s\}, V_2]$. Since G is complete, $[V_0 \setminus \{s\}, V_2] \neq \emptyset$ and T' is strictly contained in T. Thus, $x(T) \geq k$ is redundant with respect to the inequalities

$$\begin{aligned} &x(T') \geq k, \\ &x(e) \geq 0 \text{ for all } e \in [V_0 \setminus \{s\}, V_2], \end{aligned}$$

and hence cannot define a facet of kHPP(G).

2) Suppose that condition 1) holds. Let $A = [s, V_1] \cup [V_3, t] \cup [s, t]$ and let u_i be a fixed node of V_i, $i = 1, 2, 3$. Let us suppose that $\mathcal{F}$ is a facet of kHPP(G) different from a trivial inequality. Thus there exists a solution F of the kHPP such that $x^F \in \mathcal{F}$ and $F \cap [V_1, V_3] \neq \emptyset$. If this is not the case, then $\mathcal{F}$ would be equivalent to a facet defined by any of the inequalities $x(e) \geq 0$, $e \in [V_1, V_3]$. Hence, as $F \cap [V_1, V_3] \neq \emptyset$, from Lemma 4.2.2, we have that $|F \cap A| \geq k + 1$.

Sufficiency.
Suppose that conditions 1) and 2) hold. First we show that $\mathcal{F} \neq \emptyset$. As $|[s, V_1] \cup [V_3, t] \cup [s, t]| \geq k + 1$, there exist node sets $U_1 \subseteq V_1$, $U_3 \subseteq V_3$, and an edge set $E_0 \subseteq [s, t] \setminus \{st\}$ such that $|U_1| + |U_3| + |E_0| = k$. Consider the st-paths (su, ut), $u \in U_1 \cup U_3$ and (e), $e \in E_0$. Clearly, these st-paths form a set of k edge-disjoint 3-st-paths. Moreover, each of these paths intersects T only once. Thus they induce a solution, say E_1, of the kHPP whose incidence vector belongs to $\mathcal{F}$. Therefore $\mathcal{F} \neq \emptyset$.

Now suppose that there exists a facet defining inequality $bx \geq \beta$ such that $\mathcal{F} \subseteq \{x \in k\text{HPP}(G) \mid bx = \beta\}$. As before, we will show that there exists a scalar $\rho \neq 0$ such that $b = \rho a$.

Let $e \in E_1 \cap T$ (where E_1 is the solution introduced above). Let $E_2 = (E_1 \setminus \{e\}) \cup \{st\}$. Since E_2 is a solution of the kHPP whose incidence vector belongs to $\mathcal{F}$, we have

$bx^{E_2} = bx^{E_1} = \beta$, implying that $b(e) = b(st)$. As e is an arbitrary edge, we then obtain that

$$b(e) = \rho \text{ for all } e \in (E_1 \cap T) \cup \{st\}, \text{ for some } \rho \in \mathbb{R}. \tag{4.7}$$

Now let $e \in E \setminus T$. If $e \notin E_1$, then let $E_3 = E_1 \cup \{e\}$ is a solution of the kHPP. Moreover, its incidence vector belongs to $\mathcal{F}$. Hence, $b(e) = bx^{E_3} - bx^{E_1} = 0$. If $e \in E_1 \setminus T$, then e is either of the form su, $u \in U_1$, or vt, $v \in V_3$. Suppose, w.l.o.g., that $e = su$, the case where $e = vt$ is similar. Note that, by the definition of E_1, ut also belongs to E_1. Let $E'_3 = (E_1 \setminus \{su, ut\}) \cup \{st\}$. We have that E'_3 induces of the kHPP and $x^{E'_3} \in \mathcal{F}$. Hence, $bx^{E'_3} = bx^{E_1} = \beta$ and, in consequence, $b(su) + b(ut) = b(st)$. As, by (4.7), $b(ut) = b(st)$, we have that $b(su) = 0$. Thus, we obtain that

$$b(e) = 0 \text{ for all } e \in E \setminus T. \tag{4.8}$$

Consider now an edge $e \in T \setminus E_1$. If $e \in [s,t] \setminus \{st\}$, then clearly, the edge set $(E_1 \setminus \{g\}) \cup \{e\}$ induces a solution of the kHPP and its incidence vector belongs to $\mathcal{F}$ where g is an edge of E_1. Hence, as before, $b(e) = b(g) = \rho$.
Now if $e = sv$ (resp. $e = vt$) with $v \in V_2$, then the edge set $E_4 = (E_1 \setminus \{su_3\}) \cup \{e, vu_3\}$ (resp. $E_4 = (E_1 \setminus \{u_1t\}) \cup \{u_1v, e\}$) induces a solution of the kHPP. Moreover, its incidence vector belongs to $\mathcal{F}$. Thus, $bx^{E_4} - bx^{E_1} = b(e) + b(vu_3) - b(su_3) = 0$ (resp. $bx^{E_4} - bx^{E_1} = b(u_1v) + b(e) - b(u_1t) = 0$). From (4.7) and (4.8) we get $b(e) = \rho$.

Let $e = sv$ with $v \in V_3$. The case where $e \in [V_1, t]$ is similar. If $v \in U_3$, then the edge set $E_5 = (E_1 \setminus \{f\}) \cup \{e\}$, where f is the edge of E_1 between s and v, induces a solution of the kHPP whose incidence vector belongs to $\mathcal{F}$. Hence $bx^{E_5} - bx^{E_1} = b(e) - b(su_3) = 0$. By (4.7), we get $b(e) = \rho$. If $v \notin U_3$, then we have that $E'_5 = (E_1 \setminus \{f'\}) \cup \{e, vt\}$, where $f' \in E_1 \cap [s, U_3]$, also induces a solution of the kHPP and its incidence vector belongs to $\mathcal{F}$. Thus, $bx^{E'_5} - bx^{E_1} = b(e) + b(u_3t) - b(f) = 0$. By (4.7) and (4.8), we get $b(e) = \rho$.

Now suppose that $e = uv \in [V_1, V_3]$. If $u \in U_1$ and $v \in U_3$, then by considering the edge set $E_6 = (E_1 \setminus \{ut, sv\}) \cup \{e, st\}$, we get $b(e) + b(st) = b(sv) + b(ut)$. From (4.7) and (4.8), we have that $b(e) = \rho$. If $u \notin U_1$ and $v \in U_3$, then by considering the edge set $E_7 = (E_1 \setminus \{g\}) \cup \{su, e\}$, where g is the edge of E_1 between s and v, we get $b(e) + b(su) = b(g)$. By (4.7) and (4.8), we have $b(e) = \rho$. If $u \in U_1$ and $v \notin U_3$, then we show in a similar way that $b(e) = \rho$. If $u \notin U_1$ and $v \notin U_3$, then by considering the edge set $E_8 = (E_1 \setminus \{st\}) \cup \{su, e, vt\}$, we get $b(e) = \rho$. Thus, we obtain

$$b(e) = \rho \text{ for all } e \in T \setminus (E_1 \cup \{st\}). \tag{4.9}$$

From (4.7), (4.8) and (4.9), we have

$$b(e) = \begin{cases} \rho & \text{for all } e \in T, \\ 0 & \text{if not.} \end{cases}$$

Therefore, $b = \rho a$. Moreover $\rho \neq 0$ since $bx \geq \beta$ defines a facet which ends the proof of the theorem. □

As it will turn out in the next section, the conditions given for inequalities (4.1)-(4.3) to define facets will be useful for characterizing the kHPP polytope.

4.3 Complete description of kHPP(G)

In this section, we will present our main result, that is the polytope $P_k(G)$, given by the st-cut, the 3-path-cut and the trivial inequalities, is integral, which implies that kHPP(G) is completely described by these inequalities.

To this end, consider an undirected graph $G = (V, E)$. Let $N = V \setminus \{s, t\}$, N' be a copy of N and $\widetilde{V} = N \cup N' \cup \{s, t\}$. The copy in N' of a node $u \in N$ will be denoted by u'. Let $\widetilde{G} = (\widetilde{V}, \widetilde{A})$ be the directed graph such that $\widetilde{V} = N \cup N' \cup \{s, t\}$ and $\widetilde{A}$ is obtained from as follows. To each edge $e \in [s, t]$, we associate an arc from s to t in $\widetilde{G}$. To each edge $su \in E$ (resp. $vt \in E$), we associate in $\widetilde{G}$ the arc (s, u), $u \in N$ (resp. (v', t), $v' \in N'$). To each edge $uv \in E$, with $u, v \notin \{s, t\}$, we associate two arcs (u, v') and (v, u'), with $u, v \in N$ and $u', v' \in N'$. Finally, to each node $u \in V \setminus \{s, t\}$, we associate in $\widetilde{G}$ k arcs (u, u') (see Figure 4.2 for an illustration for $k = 3$).

Remark that any st-dipath in $\widetilde{G}$ is of length no more than 3. Also note that each 3-st-path in G corresponds to an st-path in $\widetilde{G}$ and vice-versa. In fact, a 3-st-path $\Gamma = (s, u, v, t)$, with $u \neq v$, $u, v \notin \{s, t\}$, corresponds to an st-path in $\widetilde{G}$ of the form (s, u, v', t) with $u \in N$ and $v' \in N'$, and a 3-st-path $L = (s, u, t)$, $u \notin \{s, t\}$ corresponds to an st-path in $\widetilde{G}$ of the form (s, u, u', t).

The main idea of the proof is to show that each solution of $P_k(G)$ corresponds to a solution of kADPP$(\widetilde{G})$ and vice versa. We will use this correspondance together with Theorem 4.1.4 to achieve the proof.

Given a solution $\overline{x}$ of $\mathbb{R}^E$, we let $\overline{y}$ be the solution of $\mathbb{R}^{\widetilde{A}}$ obtained from $\overline{x}$ as follows.

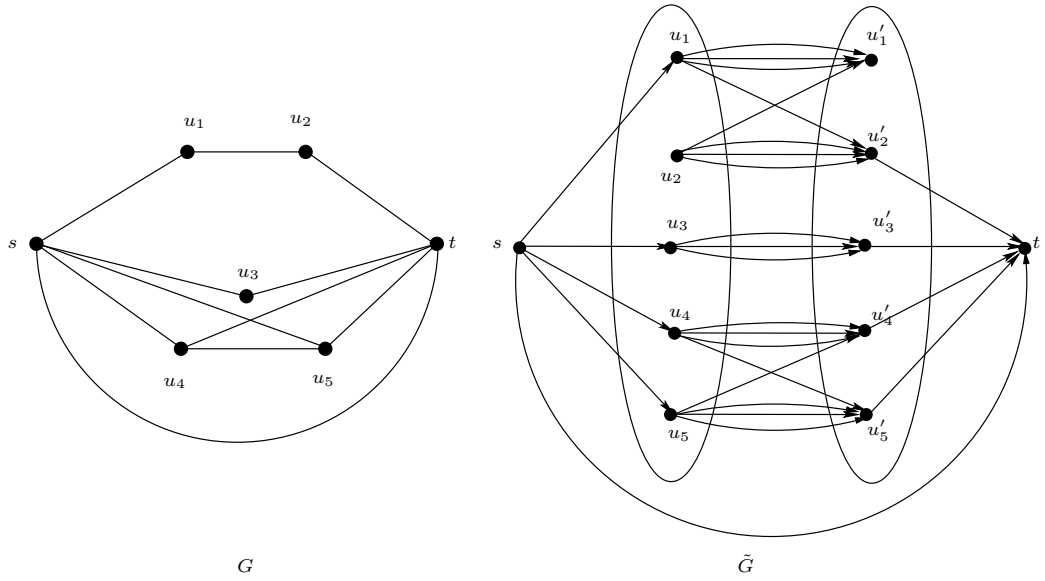


Figure 4.2: Construction of $\widetilde{G}$

$$\overline{y}(a) = \begin{cases} \overline{x}(su) & \text{if } a = (s,u),\ u \in N, \\ \overline{x}(vt) & \text{if } a = (v',t),\ v' \in N', \\ \overline{x}(uv) & \text{if } a \in \{(u,v'),(v',u)\},\ u,v \in N,\ u',v' \in N',\ u \neq v,\ u' \neq v', \\ \overline{x}(st) & \text{if } a = (s,t), \\ 1 & \text{if } a = (u,u'),\ u \in N,\ u' \in N'. \end{cases}$$

We will say that the solutions $\overline{x}$ and $\overline{y}$ are *associated*.

In what follows we will show that any st-cut and 3-path-cut of G corresponds to an st-dicut in $\widetilde{G}$. Indeed, let us consider an edge set $C \subseteq E$ and an arc set $\widetilde{C} \subseteq \widetilde{A}$ obtained from C as follows.

i) For an edge $st \in C$, add (s,t) in $\widetilde{C}$;

ii) for an edge $su \in C$, add (s,u) in $\widetilde{C}$, $u \in N$;

iii) for an edge $vt \in C$, add (v',t) in $\widetilde{C}$, $v' \in N'$;

iv) for an edge $uv \in C$, $u \neq v$, $u,v \in N$,

iv.1) if $su \in C$ or $vt \in C$, then add (v,u') in $\widetilde{C}$, with $v \in N$ and $u' \in N'$;

iv.2) if $su \notin C$ and $vt \notin C$, then add (u, v') in $\widetilde{C}$.

Observe that $\widetilde{C}$ does not contain any arc of the form (u, u') with $u \in N$ and $u' \in N'$. Also note that $\widetilde{C}$ does not contain at the same time two arcs (u, v') and (v, u'), for an edge $uv \in E$ with $u, v \in V \setminus \{s, t\}$.

Conversly, an arc subset $\widetilde{C}$ of $\widetilde{A}$ can be obtained from an edge set $C \subseteq E$ if $\widetilde{C}$ does not contain simultaneously two arcs (u, v') and (v, u'), $u, v \in N$, $u', v' \in N'$, and does not contain any arc of the form (u, u') with $u \in N$, $u' \in N'$.

As each arc of C corresponds to a single arc of $\widetilde{C}$ and vice versa, both sets have the same weight, that is $\overline{x}(C) = \overline{y}(\widetilde{C})$.

Lemma 4.3.1 *Let $C \subseteq E$ be an edge set of G which is an st-cut or a 3-path-cut induced by a partition $(V_0, ..., V_4)$ such that $|V_0| = |V_4| = 1$. Then the arc set obtained from C by the procedure given above is an st-dicut of $\widetilde{G}$. Moreover, $\overline{x}(C) = \overline{y}(\widetilde{C})$*

Proof. Suppose first that C is an st-cut $\delta(W)$ for some $W \subset V$ with $s \in W$ and $t \in \overline{W}$. Let $\widetilde{W} \subseteq \widetilde{V}$ such that $\widetilde{W} = W \cup \{u' \mid u \in W \setminus \{s\}\}$. We will show that $\widetilde{C} = \delta^+(\widetilde{W})$. We first show that $\widetilde{C} \subseteq \delta^+(\widetilde{W})$. Observe that any arc f of $\widetilde{C}$ is of the form (s, t), (s, u), $u \neq t$, (v', t), (u, v') or (v, u'), $u, v \in N$, $u', v' \in N'$. In fact, if $f = (s, u) \in \widetilde{C}$, with $u \in N \cup \{t\}$, then $su \in C$. Thus, $u \in \overline{W}$ and therefore, $(s, u) \in \delta^+(\widetilde{W})$.

If $f = (v', t)$ for $v' \in N'$, this implies that $vt \in C$. Thus, $v \in W$ and hence $(v', t) \in \delta^+(\widetilde{W})$.

If $f = (v, u')$ for $v \in N$, $u' \in N'$, then by step iv.a) of the construction of $\widetilde{C}$, we should have su and vt in C. Hence, $v \in W$ and $u \in \overline{W}$. Therefore, $v \in \widetilde{W}$ and $u' \in \widetilde{V} \setminus \widetilde{W}$. Hence $(v, u') \in \delta^+(\widetilde{W})$. If $f = (u, v')$, it similarly follows that $f \in \delta^+(\widetilde{W})$. Consequeltly, we have that $\widetilde{C} \subseteq \delta^+(\widetilde{W})$.

Next, we show that $\delta^+(\widetilde{W}) \subseteq \widetilde{C}$. Let g be an arc of $\delta^+(\widetilde{W})$. If $g = (s, u)$ for $u \in N$, then $u \in \widetilde{V} \setminus \widetilde{W}$ and hence $su \in \delta(W)(= C)$. This implies that $(s, u) \in \widetilde{C}$.

If $g = (v', t)$ for $v' \in N'$, then v' and hence v belongs to $\widetilde{W}$. Thus, $vt \in \delta(W)$ and therefore $(v', t) \in \widetilde{C}$. If $g = (v, u')$ with $v \in N$ and $u' \in N'$, then $v \in \widetilde{W}$, and $u, u' \in \widetilde{V} \setminus \widetilde{W}$. This implies that $v \in W$ and $u \in \overline{W}$. In consequence, $su \in \delta(W)$ and $vt \in \delta(W)$, and thus $(v, u') \in \widetilde{C}$.

If $g = (u, v')$ with $u \in N$ and $v' \in N'$, we similarly show that $g \in \widetilde{C}$.

We thus obtain that $\delta^+(\widetilde{W}) \subseteq \widetilde{C}$, and hence $\delta^+(\widetilde{W}) = \widetilde{C}$.

Now suppose that C is a 3-path-cut induced by a partition $(V_0, V_1, V_2, V_3, V_4)$ such that $V_0 = \{s\}$ and $V_4 = \{t\}$. By considering $\widetilde{W} = V_1 \cup \{u' \mid u \in V_1 \cup V_2\}$, we can show as before that $\widetilde{C} = \delta^+(\widetilde{W})$. □

Note that for an edge set C which is a 3-path-cut of G, induced by a partition $(V_0, ..., V_4)$ such that $|V_0| \geq 2$ or $|V_4| \geq 2$, the corresponding arc set $\widetilde{C}$ may not be an st-dicut of $\widetilde{G}$. In fact, $\widetilde{C}$ may simultaneously contain two arcs $(s, u), (u, v')$ or $(u, v'), (v', t)$. In the example of Figure 4.3, $\widetilde{C}$ simultaneously contains the arcs (s, u_2) and (u_2, u_0'). If there exists a node subset $\widetilde{W} \subseteq \widetilde{V}$ such that $\widetilde{C} = \delta^+(\widetilde{W})$, we would have $u_2 \in \widetilde{W}$ and $u_2 \in \widetilde{V} \setminus \widetilde{W}$, a contradiction.

Also note that by Theorem 4.2.4, the L-path-cut inequalities induced by such partitions do not define facets of kHPP(G).

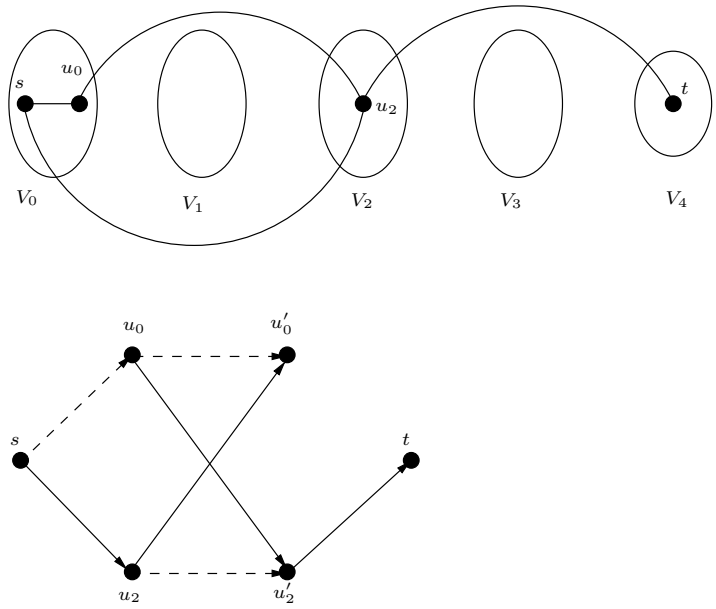


Figure 4.3: A 3-path-cut in G which does not induce an st-dicut in $\widetilde{G}$.

The following lemma shows that an st-dicut in $\widetilde{G}$ which does not contain any arc of the form (u, u'), $u \in V \setminus \{s, t\}$ corresponds to either an st-cut or a 3-path-cut in G with a lower weight.

Lemma 4.3.2 *Let $\widetilde{C}$ be an st-dicut of $\widetilde{G}$ such that $\widetilde{C}$ does not contain an arc of the*

form (u, u'), $u \in V \setminus \{s, t\}$. *Then there exists an st-cut or a 3-path-cut* $C \subseteq E$ *in* G *such that* $\overline{x}(C) \leq \overline{y}(\widetilde{C})$.

Proof. Let $\widetilde{C} = \delta^+(\widetilde{W})$ with $\widetilde{W} \subset \widetilde{V}$. Since $\widetilde{C}$ does not contain any arc of the form (u, u'), $u \in N$, $\widetilde{C}$ may contain arcs of the form either (u, v') or (v, u') or none of them but not both.

If $\widetilde{C}$ contains an arc of the form (u, v') with $u \in N$, $v' \in N'$, since $\widetilde{C}$ is an st-dicut in $\widetilde{G}$, the arcs (s, u) and (v', t) are not in $\widetilde{C}$. If $\widetilde{C}$ contains an arc (v, u'), as $\widetilde{C}$ does not contain arcs of the form (z, z'), $z \in N$, we should have $u \in \widetilde{V} \setminus \widetilde{W}$ and $v' \in \widetilde{W}$. Hence (s, u) and (v', t) are in $\widetilde{C}$. Therefore $\widetilde{C}$ can be obtained from an edge set $C \subseteq E$ of G. Moreover $\overline{x}(C) = \overline{y}(\widetilde{C})$.

Futhermore, C intersects all the 3-st-paths of G. In fact, if there exists a 3-st-path $\Gamma = (su, uv, vt)$ which does not intersect C, then the arcs (s, u), (u, v'), (v, u') and (v', t) of $\widetilde{G}$ are not in $\widetilde{C}$. Thus, the st-path $((s, u), (u, v'), (v', t))$ of $\widetilde{G}$ does not intersect $\widetilde{C}$, contradicting the fact that $\widetilde{C}$ is an st-dicut of $\widetilde{G}$. Thus C intersects all the 3-st-paths of G.

If C is an st-cut then the result holds. If this is not the case, then we will show that there exists a 3-path-cut T such that $T \subseteq C$. Consider the graph G' obtained from G by deleting all the edges of C. G' does not contain any 3-st-path since C intersets all these paths. Let $\pi = (V_0, ..., V_4)$ be a partition of V in G' such that $V_0 = \{s\}$, V_i, for $i = 1, 2, 3$, is the set of nodes of G' at distance (in terms of edges) i from s and $V_4 = V \setminus (\bigcup_{i=0}^{3} V_i)$. As C intersects all the 3-st-paths of G, all the st-paths in G' are of length at least 4 and hence, $t \in V_4$. Moreover, the subgraph G'_π induced by π in G' does not contain any chord, that is an edge uv with $u \in V_i$, $v \in V_j$, and $|i - j| > 1$. In fact, if uv is a chord, then v is at distance $i + 1 < j$ of s, a contradiction. Therefore, if T is the 3-path-cut induced by π, we have that $T \subseteq C$. As $\overline{x}(e) \geq 0$, for all $e \in E$, this implies $\overline{x}(T) \leq \overline{x}(C) = \overline{y}(\widetilde{C})$. □

In what follows, we will show that $P_k(G)$ is integral. To this end, we give some lemmas.

Lemma 4.3.3 *Let* $\overline{x} \in P_k(G)$ *and* $\overline{y}$ *be its associated solution. Then* $\overline{y} \in kADPP(\widetilde{G})$.

Proof. Clearly, $\overline{y}$ satisfies inequalities $0 \leq y(a) \leq 1$, for all $a \in \widetilde{A}$. Now suppose that there exists an st-dicut inequality, say $y(\delta^+(\widetilde{W})) \geq k$ with $\widetilde{W} \subseteq \widetilde{V}$, such that $\overline{y}(\delta^+(\widetilde{W})) < k$.

First note that $\delta^+(\widetilde{W})$ does not contain any arc of the form (u, u'), $u \in N$. In fact, if $(u, u') \in \delta^+(\widetilde{W})$, for some $u \in N$, then one would have that $[u, u'] \subseteq \delta^+(\widetilde{W})$. Since $|[u, u']| = k$ and $\overline{y}(a) = 1$ for all $a \in [u, u']$, one would have $\overline{y}(\delta^+(\widetilde{W})) \geq k$, a contradiction. Hence, from Lemma 4.3.2, there exists either an st-cut or a 3-path-cut $C \subseteq E$ of G such that $\overline{x}(C) \leq \overline{y}(\delta^+(\widetilde{W}))$ and therefore $\overline{x}(C) < k$. But this is impossible since $\overline{x} \in P_k(G)$. □

Lemma 4.3.4 *Let $e = uv$ be an edge of G such that $u, v \in V \setminus \{s, t\}$, and $\overline{y} \in \mathbb{R}^{\widetilde{A}}$ a solution of kADPP($\widetilde{G}$). If there exists an st-dicut $\widetilde{C}$ of $\widetilde{G}$ which does not contain any arc of the form (z, z'), $z \in V \setminus \{s, t\}$, and such that $(u, v') \in \widetilde{C}$ and $\overline{y}(\widetilde{C}) = k$, then $\overline{y}(\widetilde{C}') > k$ for all st-dicut $\widetilde{C}'$ of $\widetilde{G}$ containing the arc (v, u').*

Proof. Suppose that there exists an st-dicut $\widetilde{C} = \delta^+(\widetilde{W})$ of $\widetilde{G}$ which does not contain arcs of the form (z, z'), $z \in V \setminus \{s, t\}$ and such that $(u, v') \in \widetilde{C}$ and $\overline{y}(\widetilde{C}) = k$. Suppose also, on the contrary, that there exists an st-dicut $\widetilde{C}' = \delta^+(\widetilde{W}')$ containing the arc (v, u') and such that $\overline{y}(\widetilde{C}') = k$. From Theorem 4.1.5, $\widetilde{W}$ and $\widetilde{W}'$ can be chosen so that either $\widetilde{W}' \subseteq \widetilde{W}$ or $\widetilde{W} \subseteq \widetilde{W}'$. As $(u, v') \in \widetilde{C}$, we have that $u \in \widetilde{W}$ and $v' \in \widetilde{V} \setminus \widetilde{W}$. Since $(z, z') \notin \widetilde{C}$, for all $z \in V \setminus \{s, t\}$, it follows that $u, u' \in \widetilde{W}$, and $v, v' \in \widetilde{V} \setminus \widetilde{W}$. Similarly, as $(v, u') \in \widetilde{C}'$, we have that $v, v' \in \widetilde{W}'$ and $u, u' \in \widetilde{V} \setminus \widetilde{W}'$.

If $\widetilde{W}' \subseteq \widetilde{W}$, then one would have $v \in \widetilde{W}$. But this contradicts the fact that $v \in \widetilde{V} \setminus \widetilde{W}$. If $\widetilde{W} \subseteq \widetilde{W}'$, then we would obtain that $u \in \widetilde{W}'$. As $u \in \widetilde{V} \setminus \widetilde{W}'$, this is a contradiction. □

Now we are ready to state our main result.

Theorem 4.3.1 *The polytope kHPP(G) is completely described by inequalities (4.1)-(4.3).*

Proof. We will show that the polytope $P_k(G)$ is integral. For this, let us suppose, on the contrary, that there exists a fractional extreme point $\overline{x}$ of $P_k(G)$. Then there exists a set of st-cuts $C^*(\overline{x})$ and a set of 3-path-cuts $T^*(\overline{x})$ such that $\overline{x}$ is the unique solution of the system

$$S(\overline{x}) \begin{cases} x(e) = 0, & \text{for all } e \in E_0(\overline{x}), \\ x(e) = 1, & \text{for all } e \in E_1(\overline{x}), \\ x(C) = k, & \text{for all } C \in C^*(\overline{x}), \\ x(T) = k, & \text{for all } T \in T^*(\overline{x}), \end{cases}$$

where $E_0(\overline{x})$ (resp. $E_1(\overline{x})$) is the set of edges such that $\overline{x}(e) = 0$ (resp. $\overline{x}(e) = 1$) and $|E_0(\overline{x})| + |E_1(\overline{x})| + |C^*(\overline{x})| + |T^*(\overline{x})| = |E|$.

We will show that there exists a solution $\overline{x}'_1$ of $P_k(G)$ different from $\overline{x}$ which is also a solution of $S(\overline{x})$, yielding a contradiction.

Clearly, the solution $\overline{y}$, associated with $\overline{x}$, is fractional and, by Lemma 4.3.3, is a solution of kADPP$(\widetilde{G})$. Let $\widetilde{A}_0(\overline{y}) = \{(u,v) \in \widetilde{A} \mid \overline{x}(uv) = 0\}$ and $\widetilde{A}_1(\overline{y}) = \{(u,v) \in \widetilde{A} \mid \overline{x}(uv) = 1\} \cup \{(u,u'),\ u \in N,\ u' \in N'\}$. By Lemma 4.3.1, each st-cut $C \in C^*(\overline{x})$ and 3-path-cut $T \in T^*(\overline{x})$ corresponds to an st-dicut $\widetilde{C}$ of $\widetilde{G}$ having the same weight, that is $\overline{y}(\widetilde{C}) = k$. We denote by $C^*(\overline{y})$ the set of the corresponding st-dicuts. It then follows that $\overline{y}$ is solution (not necessarily unique) of the system $S(\overline{y})$ given by

$$S(\overline{y}) \begin{cases} y(a) = 0, & \text{for all } a \in \widetilde{A}_0(\overline{y}), \\ y(a) = 1, & \text{for all } a \in \widetilde{A}_1(\overline{y}), \\ y(\widetilde{C}) = k, & \text{for all } \widetilde{C} \in C^*(\overline{y}). \end{cases}$$

Since $\overline{y}$ is fractional and hence, by Theorem 4.1.4, cannot be an extreme point of kADPP$(\widetilde{G})$, $\overline{y}$ can be written as a convexe combination of integral extreme points of kADPP$(\widetilde{G})$. Let $\overline{y}_1$ be one of these extreme points. Clearly, $\overline{y}_1$ is also a solution of $S(\overline{y})$. In the following, we show that there exists an integer solution $\overline{y}'_1$ of kADPP$(\widetilde{G})$ which is a solution of $S(\overline{y})$ and such that $\overline{y}'_1(u,v') = \overline{y}'_1(v,u')$ for all pair of arcs $((u,v'),(v,u'))$ of $\widetilde{G}$, corresponding to an edge $uv \in E$ with $u,v \in V \setminus \{s,t\}$ and $u \neq v$. If such a solution exists, then $\overline{y}'_1$ can be associated with a solution $\overline{x}'_1 \in P_k(G)$ satisfying $S(\overline{x})$ and different from $\overline{x}$.

If for all pair of arcs $((u,v'),(v,u'))$ of $\widetilde{G}$, with $u,v \in N$, $u',v' \in N'$, $\overline{y}_1(u,v') = \overline{y}_1(v,u')$, then we can take $\overline{y}'_1 = \overline{y}_1$. So suppose that there exist two nodes $u,v \in V \setminus \{s,t\}$, such that $uv \in E$ and $\overline{y}_1(u,v') \neq \overline{y}_1(v,u')$. As $\overline{y}_1$ is integral, we can suppose, w.l.o.g., that $\overline{y}_1(u,v') = 1$ and $\overline{y}_1(v,u') = 0$. It follows that $\overline{y}(u,v')$, $\overline{y}(v,u')$, $\overline{x}(uv)$ are fractional. Note that $\overline{x}(uv) = \overline{y}(u,v') = \overline{y}(v,u')$. Also note that any st-dicut of $\widetilde{G}$ inducing a tight st-dicut inequality for $\overline{y}$ or $\overline{y}_1$ does not contain arcs of the form (z,z'), $z \in V \setminus \{s,t\}$. If there is an st-dicut $\widetilde{C}$ of $\widetilde{G}$ which contains (u,v'), and such that $\overline{y}_1(\widetilde{C}) = k$, then, by Lemma 4.3.4, every st-dicut containing (v,u') is not tight for

$\overline{y}_1$. Let $\overline{y}'_1$ be the solution given by

$$\overline{y}'_1(a) = \begin{cases} \overline{y}_1(a), & \text{for all } a \in \widetilde{A} \setminus \{(v,u')\}, \\ 1, & \text{for } a = (v,u'). \end{cases}$$

Clearly, $\overline{y}'_1$ is a solution of kADPP$(\widetilde{G})$ with $\overline{y}'_1(u,v') = \overline{y}'_1(v',u) = 1$, and satisfies with equality every st-dicut inequality which is tight for $\overline{y}_1$. In particular, the st-dicuts inequalities of $\widetilde{C}^*(\overline{y})$ are also tight for $\overline{y}'_1$. Hence, $\overline{y}'_1$ is a solution of $S(\overline{y})$.

If there is an st-dicut $\widetilde{C}$ which contains (v,u') and such that $\overline{y}_1(\widetilde{C}) = k$, then, by Lemma 4.3.4, every st-dicut $\widetilde{R} \subseteq \widetilde{A}$ containing (u,v') is such that $\overline{y}_1(\widetilde{R}) \geq k+1$. Hence, the solution $\overline{y}'_1$ given by

$$\overline{y}'_1(a) = \begin{cases} \overline{y}_1(a), & \text{for all } a \in \widetilde{A} \setminus \{(u,v')\}, \\ 0, & \text{for } a = (u,v'), \end{cases}$$

is a solution of kADPP$(\widetilde{G})$ such that $\overline{y}'_1(u,v') = \overline{y}'_1(v',u) = 0$, and every st-dicut inequality which is tight for $\overline{y}_1$ is also tight for $\overline{y}'_1$. Thus $\overline{y}'_1$ is also a solution of $S(\overline{y})$.

Consequently, there exists an integer solution $\overline{y}'_1 \in k$ADPP$(\widetilde{G})$ which is a solution of $S(\overline{y})$ and such that $\overline{y}'_1(u,v') = \overline{y}'_1(u',v)$ for all arcs $(u,v'),(v,u') \in \widetilde{A}$ corresponding to an edge $uv \in E$. Thus, $\overline{y}'_1$ can be associated with a solution $\overline{x}'_1$ of $P_k(G)$. As $\overline{y}'_1$ is integral, $\overline{x}'_1$ is also integral. Moreover, $\overline{x}'_1$ is a solution of $S(\overline{x})$. In fact, it is not hard to see that, as $\overline{y}'_1$ is a solution of $S(\overline{y})$, and $\overline{y}'_1(a) = 0$ for all $a \in \widetilde{A}_0(\overline{y})$ and $\overline{y}'_1(a) = 1$ for all $a \in \widetilde{A}_1(\overline{y})$. Hence $\overline{x}'_1(e) = 0$ for all $e \in E_0(\overline{x})$ and $\overline{x}'_1(e) = 1$ for all $e \in E_1(\overline{x})$. Suppose that there is an st-cut (resp. 3-path-cut) inequality in $C^*(\overline{x})$ (resp. $T^*(\overline{x})$) which is not tight for $\overline{x}'_1$, say $\overline{x}'_1(C_0) > k$. Then by Lemma 4.3.2, we have that $\overline{x}'_1(C_0) \leq \overline{y}_1(\widetilde{C}_0)$, where $\widetilde{C}_0$ is the st-dicut of $\widetilde{C}^*(\overline{y})$ corresponding to C_0. We thus obtain that $\overline{y}'_1(\widetilde{C}_0) > k$. Hence $\overline{y}'_1$ is not a solution of $S(\overline{y})$, a contradiction. Thus, $\overline{x}'_1$ is a solution of $S(\overline{x})$. Since $\overline{x}'_1$ is integral and $\overline{x}$ is fractional, $\overline{x}'_1 \neq \overline{x}$. In consequence, $\overline{x}$ is not the unique solution of $S(\overline{x})$, contradicting the fact that $\overline{x}$ is an extreme point of $P_k(G)$. Therefore, $\overline{x}$ cannot be fractional, which ends the proof of the theorem. □

A direct consequence of Theorems 4.2.2, 4.2.3, 4.2.4 and 4.3.1 is the following.

Corollary 4.3.1 *If $G = (V,E)$ is a complete graph and $|V| \geq k+2$, a minimal*

complete linear description of kHPP(G) is given by

$$
\begin{aligned}
x(\delta(W)) \geq k \quad & \textit{for all } st-\textit{cut } \delta(W), \\
x(T) \geq k \quad & \textit{for all 3-path-cut } T \textit{ induced by a partition satisfying} \\
& \textit{conditions 1) and 2) of Theorem 4.2.4}, \\
x(e) \geq 0 \quad & \textit{for all } e \in E, \\
x(e) \leq 1 \quad & \textit{for all } e \in E.
\end{aligned}
$$

As mentionned in Section 4.1.1, the separation problem for the *st*-cut and 3-path-cut inequalities can be solved in polynomial time. Thus, the kHPP can be solved in polynomial time using a cutting plane algorithm.

4.4 Concluding remarks

In this chapter we have given a complete description of the polytope associated with the k edge-disjoint hop-constrained paths problem when $L = 3$ and $k \geq 2$. We have presented valid inequalites for the problem and given an integer programming formulation. We have also described necessary and sufficient conditions for the trivial inequalities, the *st*-cut and L-path-cut inequalities to define facets of the polytope. Using these results together with a transformation of the kHPP in G into the kADPP in a directed graph $\widetilde{G}$, we have shown that the polytope kHPP(G) is completely described by the trivial, *st*-cut and 3-path cut inequalities. As the separation problem for these inequalities can be solved in polynomial time, this yields a polynomial time cutting plane algorithm to solve the problem.

These results generalize those obtained by Huygens et al. [75] and Dahl et al. [35] for $k = 2$ and $L = 2, 3$ and for $k \geq 2$ and $L = 2$, respectively. Unfortunately the linear description of the kHPP is no longer valid when $L \geq 4$. As shown by Huygens and Mahjoub [73], further inequalities are even needed for an integer programming formulation of the problem when $k = 2$ and $L = 4$.

The kHPP can also be seen as a minimum cost flow problem in the graph $\widetilde{G}$ by associated with its arcs unit capacities and appropriate weights. In fact, an arc of $\widetilde{G}$ which corresponds to an edge of G takes the same weight as this edge while the arcs of the form (u, u'), $u \in V \setminus \{s, t\}$ (which do not correspond to any edge in G) are given the weight 0. By the correspondance between the 3-*st*-paths of G and the

st-paths in $\widetilde{G}$, a minimum weight subgraph of G which contains k edge-disjoint 3-st-paths corresponds to a subgraph of $\widetilde{G}$ containing k arc-disjoint st-paths of the same weight. Moreover, the weight of this subgraph is minimum. The kHPP is thus equivalent to finding a minimum cost flow from s to t of value k in $\widetilde{G}$. This implies that the problem can also be solved in polynomial time using any minimum cost flow algorithm.

The integer programming formulation for the kHPP can be easily extended to the more general case where more than pair of terminals are considered. However, as pointed out in [74], the cut inequalities toghether with the L-path-cut and trivial inequalities do not suffice to completely describe the kHPP polytope even when only two pair of terminals are considered $L \geq 3$ and $k = 2$.

The results of this chapter can be exploited in a Branch-and-Cut algorithm for that general case. Also the transformation of the kHPP into the kADPP in an appropriate directed graph introduced and exploited here, can be used to give based flow formulations. It would also be interesting to investigate this type of approach for $L \geq 4$. This is our aim in the next chapter.

Chapter 5

The k-Edge-Connected Hop-Constrained Network Design Problem

Let $G = (V, E)$ be an undirected graph, a set of demands $D \subseteq V \times V$, a cost function $c : E \rightarrow \mathbb{R}$, which associates the cost $c(e)$ with each edge $e \in E$. The *k-Edge-Connected Hop-Constrained Network Design Problem* (kHNDP for short) consists in finding a minimum cost subgraph of G such that there exist k edge-disjoint L-st-paths between the terminals of each demand $\{s, t\}$ of D.

In this chapter, we consider the kHNDP with $L = 2, 3$ and $k \geq 2$ and introduce four new integer programming formulations for the problem. In Section 5.1, we give a formulation of the kHNDP using the design variables. In Sections 5.2 and 5.3, we introduced four new integer programming formulations. These formulations use transformations of the initial undirected graph into directed graphs.

5.1 Integer programming formulation for the kHNDP using the design variables

Let $G = (V, E)$ be an undirected graph, $L \geq 2$ and $D = \{\{s_1, t_1\}, ..., \{s_d, t_d\}\}$, $d \geq 2$, be the set of demands. We will denote by R_D the set of terminal nodes of G, that is those nodes of G which are involved in at least one demand. It is clear that the incidence

vector x^F of any solution (V, F) of the kHNDP satisfies the following inequalities.

$$x(\delta(W)) \geq k \text{ for all } st\text{-cut}, \{s,t\} \in D, \tag{5.1}$$
$$x(T) \geq k \quad \text{for all } L\text{-}st\text{-path-cut}, \{s,t\} \in D, \tag{5.2}$$
$$x(e) \geq 0 \quad \text{for all } e \in E, \tag{5.3}$$
$$x(e) \leq 1 \quad \text{for all } e \in E, \tag{5.4}$$

Conversely, any integer solution of the system defined by inequalities (5.1)-(5.4) is the incidence vector of a solution of the kHNDP when $L = 2, 3$.

Recall that inequalities (5.1), (5.2) and (5.3)-(5.4) are called respectively *st-cut inequalities*, *L-st-path-cut inequalities* and *trivial inequalities*.

It is not hard to see that the kHNDP can be formulated as a linear integer program similarly to the case of a single demand (Chapter 4). The following lemma and theorem give this result. Their proof are similar to those of Lemma 4.1.1 and Theorem 4.1.1.

Lemma 5.1.1 *Let $G = (V, E)$ be an undirected graph and s and t two nodes of V. Suppose that there do not exist k edge-disjoint L-st-paths in G, with $k \geq 2$. Then there exists a set of at most $k-1$ edges that intersects every L-st-path.*

Theorem 5.1.1 *Let $G = (V, E)$ be a graph, $k \geq 2$ and $L \in \{2, 3\}$. Then the kHNDP is equivalent to the following inter program*

$$min\{cx;\ \textit{subject to } (5.1) - (5.4),\ x \in \mathbb{Z}^E\}. \tag{5.5}$$

Formulation (5.5) will be called *Natural formulation*. We will denote it by $k\text{HNDP}_{Nat}$. It only uses the design variables.

In the next sections, we give new integer programming formulations for the kHNDP in the case where $k \geq 2$ and $L = 2, 3$.

5.2 Separated formulations for the kHNDP

In this section we introduce three integer programming formulations for the kHNDP. Let $G = (V, E)$ be an undirected graph, $L \in \{2, 3\}$, $k \geq 2$, two integers, and D a set of demands. Before giving these formulations, we introduce a graph transformation which produces $|D|$ directed graphs from the graph G.

5.2.1 Graph transformation

Let $\{s,t\} \in D$ and $\widetilde{G}_{st} = (\widetilde{V}_{st}, \widetilde{A}_{st})$ be the directed graph obtained from G using the following procedure.

Let $N_{st} = V \setminus \{s,t\}$, N'_{st} be a copy of N_{st} and $\widetilde{V}_{st} = N_{st} \cup N'_{st} \cup \{s,t\}$. The copy in N'_{st} of a node $u \in N_{st}$ will be denoted by u'. To each edge $e = st \in E$, we associate an arc (s,t) in $\widetilde{G}_{st}$ with capacity 1. With each edge $su \in E$ (resp. $vt \in E$), we associate in $\widetilde{G}_{st}$ the arc (s,u), $u \in N_{st}$ (resp. (v',t), $v' \in N'_{st}$) with capacity 1. With each node $u \in V \setminus \{s,t\}$, we associate in $\widetilde{G}_{st}$ one arc (u,u') with an infinit capacity. Finally, if $L = 3$ we associate with each edge $uv \in E \setminus \{s,t\}$, two arcs (u,v') and (v,u'), with $u,v \in N_{st}$ and $u',v' \in N'_{st}$ with capacity 1 (see Figure 5.1 for an illustration with $L = 3$).

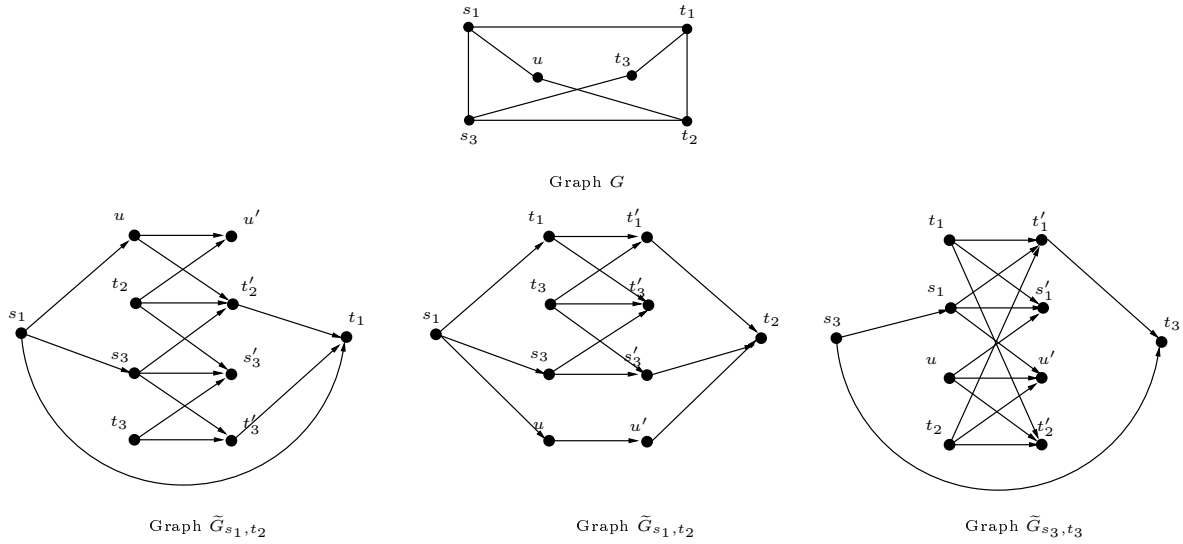


Figure 5.1: Construction of graphs $\widetilde{G}_{st}$ with $D = \{\{s_1,t_1\},\{s_1,t_2\},\{s_3,t_3\}\}$ for $L = 3$

Note that each graph $\widetilde{G}_{st}$ contains $|\widetilde{V}_{st}| = 2|V| - 2$ $(= |N_{st} \cup N'_{st} \cup \{s,t\}|)$ nodes and $|\widetilde{A}_{st}| = |\delta(s)| + |\delta(t)| - |[s,t]| + |V| - 2$ arcs if $L = 2$ and $|\widetilde{A}_{st}| = 2|E| - |\delta(s)| - |\delta(t)| + |[s,t]| + |V| - 2$ arcs if $L = 3$, for all $\{s,t\} \in D$.

Given a demand $\{s,t\}$, the associated graph $\widetilde{G}_{st} = (\widetilde{V}_{st}, \widetilde{A}_{st})$, and an edge $e \in E$, we denote by $\widetilde{A}_{st}(e)$ the set of arcs of $\widetilde{G}_{st}$ corresponding to the edge e.

It is not hard to see that $\widetilde{G}_{st}$ does not contain any circuit. Also, observe that any st-dipath in $\widetilde{G}_{st}$ is of length no more than 3. Moreover each L-st-path in G corresponds to an st-dipath in $\widetilde{G}_{st}$ and conversely. In fact, if $L \in \{2,3\}$, every 3-st-path (s,u,v,t),

with $u \neq v$, $u, v \in V \setminus \{s,t\}$, corresponds to an st-dipath in $\widetilde{G}_{st}$ of the form (s,u,v',t) with $u \in N_{st}$ and $v' \in N'_{st}$. Every 2-st-path (s,u,t), $u \in V \setminus \{s,t\}$, corresponds to an st-dipath in $\widetilde{G}_{st}$ of the form (s,u,u',t).

We have the following lemma.

Lemma 5.2.1 *Let $L \in \{2,3\}$ and $\{s,t\} \in D$.*

i) If two L-st-paths of G are edge-disjoint, then the corresponding st-dipaths in $\widetilde{G}_{st}$ are arc-disjoint.

ii) If two st-dipaths of $\widetilde{G}_{st}$ are arc-disjoint, then the corresponding st-paths in G contain two edge-disjoint L-st-paths.

Proof. We will suppose, w.l.o.g., that $L = 3$. The proof is similar for $L = 2$.

i) Let P_1 and P_2 be two edge-disjoint 3-st-paths of G. Let $\widetilde{P}_1$ and $\widetilde{P}_2$ be the two st-dipaths of $\widetilde{G}_{st}$ corresponding to P_1 and P_2, respectively. We will show that $\widetilde{P}_1$ and $\widetilde{P}_2$ are arc-disjoint. Let us assume that this is not the case. Then they intersect on an arc a of the form either (s,t), (s,u), (v',t), (u,v') or (u,u'), with $u \in N_{st}$ and $v' \in N'_{st}$. If a is of the form (s,t), (s,u), (v',t) or (u,v'), then it corresponds to an edge e of G of the form either st, su, vt or uv. This implies that P_1 and P_2 contain both the edge e, a contradiction. Thus, $\widetilde{P}_1$ and $\widetilde{P}_2$ intersect on an arc of the form (u,u'), with $u \in N_{st}$. As the st-dipaths of $\widetilde{G}_{st}$ contain at most 3 arcs, $\widetilde{P}_1$ and $\widetilde{P}_2$ are of the form (s,u,u',t). But this implies that P_1 and P_2 contain simulataneously the edges su and ut, a contradiction.

ii) Now consider two arc-disjoint st-dipaths $\widetilde{P}_1$ and $\widetilde{P}_2$ of $\widetilde{G}_{st}$ and let P_1 and P_2 be the corresponding 3-st-paths of G. Suppose that $P_1 \cap P_2 \neq \emptyset$. If P_1 and P_2 intersect on an edge $e = st$, then $\widetilde{P}_1$ and $\widetilde{P}_2$ also contain the arc (s,t), a contradiction. If P_1 and P_2 intersect on an edge of the form su, $u \in V \setminus \{s,t\}$ (resp. vt, $v \in V \setminus \{s,t\}$), then, as before, both $\widetilde{P}_1$ and $\widetilde{P}_2$ contain the arc (s,u) (resp. (v',t)), yielding a contradiction. Now if P_1 and P_2 intersect on an edge of the form uv, $u, v \in V \setminus \{s,t\}$, then $\widetilde{P}_1$ and $\widetilde{P}_2$ contain the arcs (u,v') and (v,u') of $\widetilde{G}_{st}$. Since $\widetilde{P}_1$ and $\widetilde{P}_2$ are arc-disjoint, $\widetilde{P}_1$ contains say (u,v') and $\widetilde{P}_2$ (v,u'). Thus they are respectively of the form (s,u,v',t) and (s,v,u',t). This implies that $P_1 = (su, uv, vt)$ and $P_2 = (sv, vu, ut)$. Let $P'_1 = (su, ut)$ and $P'_2 = (sv, vt)$. Clearly P'_1 and P'_2 are edge-disjoint and of length 2. Thus, we associate $\widetilde{P}_1$ and $\widetilde{P}_2$ with them, which ends the proof of the lemma. □

As a consequence of Lemma 5.2.1, for $L \in \{2,3\}$ and every demand $\{s,t\} \in D$, a set of k edge-disjoint L-st-paths of G corresponds to a set of k arc-disjoint st-dipaths of $\widetilde{G}_{st}$, and k arc-disjoint st-dipaths of $\widetilde{G}_{st}$ correspond to k edge-disjoint L-st-paths of G. Therefore we have the following corollary.

Corollary 5.2.1 *Let H be a subgraph of G and $\widetilde{H}_{st}$, $\{s,t\} \in D$, the subgraph of $\widetilde{G}_{st}$ obtained by considering all the arcs of $\widetilde{G}_{st}$ corresponding to an edge of H, plus the arcs of the form (u,u'), $u \in V \backslash \{s,t\}$. Then H induces a solution of the kHNDP if $\widetilde{H}_{st}$ contains k arc-disjoint st-dipaths, for every $\{s,t\} \in D$. Conversly, given a set of subgraphs $\widetilde{H}_{st}$ of $\widetilde{G}_{st}$, $\{s,t\} \in D$, if H is the subgraph of G obtained by considering all the edges of G associated with at least one arc in a subgraph $\widetilde{H}_{st}$, then H induces a solution of the kHNDP only if $\widetilde{H}_{st}$ contains k arc-disjoint st-dipaths, for every $\{s,t\} \in D$.*

Remark that a graph $\widetilde{G}_{st}$ will contain k arc-disjoint st-dipaths if and only if every st-dicut contains at least k arcs. This implies, by the Max flow - Min cut Theorem, that $\widetilde{G}_{st}$ contains k arc-disjoint st-dipaths if and only if there exists a feasible flow of value $\geq k$ where every arc of $\widetilde{G}_{st}$ has capacity 1. Given a demand $\{s,t\}$ and a feasible flow f of value $\geq k$ on $\widetilde{G}_{st}$, we will denote by $\widetilde{H}^f_{st}$ the set of arcs of $\widetilde{G}_{st}$ having a positive value of flow with respect to f.

In what follows, we will give three integer programming formulations for the kHNDP using graphs $\widetilde{G}_{st}$, $\{s,t\} \in D$. These formulations will be called *separated formulations*.

5.2.2 Cut formulation

The first formulation is based on cuts in the graphs $\widetilde{G}_{st}$, $\{s,t\} \in D$. Given a subgraph $\widetilde{H}_{st}$ of $\widetilde{G}_{st}$, we let $y^{\widetilde{H}_{st}}_{st} \in \mathbb{R}^{\widetilde{A}_{st}}$ be the incidence vector of $\widetilde{H}_{st}$, that is to say $y^{\widetilde{H}_{st}}_{st}(a) = 1$ if $a \in \widetilde{H}_{st}$ and $y^{\widetilde{H}_{st}}_{st}(a) = 0$ if not. By Corollary 5.2.1, if a subgraph H of G induces a solution of the kHNDP, then the subgraph $\widetilde{H}_{st}$ contains at least k arc-disjoint st-dipaths, for all $\{s,t\} \in D$, and conversely. Thus, for any solution H of the kHNDP, the following inequalities are satisfied by $y^{\widetilde{H}_{st}}_{st}$, for all $\{s,t\} \in D$,

$$y_{st}(\delta^+(\widetilde{W})) \geq k, \text{ for all } st\text{-dicut } \delta^+(\widetilde{W}) \text{ of } \widetilde{G}_{st}, \tag{5.6}$$

$$y_{st}(a) \leq x(e), \text{ for all } a \in \widetilde{A}_{st}(e),\ e \in E, \tag{5.7}$$

$$y_{st}(a) \geq 0, \text{ for all } a \in \widetilde{A}_{st}, \tag{5.8}$$

$$x(e) \leq 1, \text{ for all } e \in E. \tag{5.9}$$

where $y_{st} \in \mathbb{R}^{\widetilde{A}_{st}}$ for all $\{s,t\} \in D$ and $x \in \mathbb{R}^E$.

Inequalities (5.6) will be called *directed st-cut inequalities* or *st-dicut inequalities* and inequalities (5.7) *linking inequalities*. Inequalities (5.7) indicate that an arc $a \in \widetilde{A}_{st}$ corresponding to an edge e is not in $\widetilde{H}_{st}$ if e is not taken in H. Inequalities (5.8) and (5.9) are called *trivial inequalities*.

We have the following result which is given without proof since it easily follows from the above results.

Theorem 5.2.1 *The kHNDP for $L = 2, 3$ is equivalent to the following integer program*

$$\begin{aligned} min\{cx;\ &subject\ to\ (5.6) - (5.9),\ x \in \mathbb{Z}_+^E,\ y_{st} \in \mathbb{Z}_+^{\widetilde{A}_{st}},\\ &for\ all\ \{s,t\} \in D\}. \end{aligned} \tag{5.10}$$

This formulation is called *Cut formulation* and denoted by kHNDP$_{Cu}$. It contains

$$|E| + \sum_{\{s,t\}\in D} |\widetilde{A}_{st}| = |E| + d(n-2) + \sum_{\{s,t\}\in D} |\delta(s)| + \sum_{\{s,t\}\in D} |\delta(t)| - \sum_{\{s,t\}\in D} |[s,t]|$$

variables if $L = 2$ and

$$|E| + \sum_{\{s,t\}\in D} |\widetilde{A}_{st}| = |E| + 2d|E| + d(n-2) - \sum_{\{s,t\}\in D} |\delta(s)| - \sum_{\{s,t\}\in D} |\delta(t)| + \sum_{\{s,t\}\in D} |[s,t]|$$

variables if $L = 3$ (remind that $d = |D|$).

However, the number of constraints is exponential since the directed st-cuts are in exponential number in $\widetilde{G}_{st}$, for all $\{s,t\} \in D$. As it will turn out in Chapter 6, its linear relaxation can be solved in polynomial time using a cutting plane algorithm.

5.2.3 Node-Arc formulation

Let $H \subseteq E$ be a subgraph of G and x^H its incidence vector. Given a demand $\{s,t\}$, we let $f^{st} \in \mathbb{R}^{\widetilde{A}_{st}}$ be an integer flow vector on $\widetilde{G}_{st}$ of value k. Then f^{st} satisfies the *flow conservation constraints*, given by

$$\sum_{a\in\delta^+(u)} f_a^{st} - \sum_{a\in\delta^-(u)} f_a^{st} = \left\{ \begin{array}{ll} k & \text{if } u = s, \\ 0 & \text{if } u \in \widetilde{V}_{st} \setminus \{s,t\}, \\ -k & \text{if } u = t, \end{array} \right\}, \quad \text{for all } u \in \widetilde{V}_{st}. \tag{5.11}$$

Also x^H and $(f^{st})_{\{s,t\}\in D}$ satisfy the following inequalities

$$f_a^{st} \leq x(e), \text{ for all } a \in \widetilde{A}_{st}(e),\ \{s,t\} \in D,\ e \in E, \tag{5.12}$$

$$f_a^{st} \geq 0, \text{ for every } a \in \widetilde{A}_{st} \text{ and } \{s,t\} \in D, \tag{5.13}$$

$$x(e) \leq 1, \text{ for all edge } e \in E. \tag{5.14}$$

Inequalities (5.12) are also called *linking inequalities*. They indicate that if an edge $e \in E$ is not in the solution, then the flow on every arc corresponding to e is 0. Inequalities (5.13)-(5.14) are called *trivial inequalities*.

We have the following theorem which will be given without proof.

Theorem 5.2.2 *The kHNDP for $L = 2, 3$ is equivalent to the following integer program*

$$min\{cx;\ subject\ to\ (5.11) - (5.14),\ x \in \mathbb{Z}_+^E,\ f^{st} \in \mathbb{Z}_+^{\widetilde{A}_{st}},\ for\ all\ \{s,t\} \in D\}. \tag{5.15}$$

This formulation will be called *Node-Arc formulation* and denoted by $k\text{HNDP}_{NA}$. It contains

$$|E| + \sum_{\{s,t\}\in D} |\widetilde{A}_{st}| = |E| + d(n-2) + \sum_{\{s,t\}\in D} |\delta(s)| + \sum_{\{s,t\}\in D} |\delta(t)| - \sum_{\{s,t\}\in D} |[s,t]|$$

variables if $L = 2$ and

$$|E| + \sum_{\{s,t\}\in D} |\widetilde{A}_{st}| = |E| + 2d|E| + d(n-2) - \sum_{\{s,t\}\in D} |\delta(s)| - \sum_{\{s,t\}\in D} |\delta(t)| + \sum_{\{s,t\}\in D} |[s,t]|$$

variables if $L = 3$.

The number of constraints is

$$d|V| + \sum_{\{s,t\}\in D} |\delta(s)| + \sum_{\{s,t\}\in D} |\delta(t)| - \sum_{\{s,t\}\in D} |[s,t]|$$

if $L = 2$ and

$$d|V| + 2d|E| - \sum_{\{s,t\}\in D} |\delta(s)| - \sum_{\{s,t\}\in D} |\delta(t)| + \sum_{\{s,t\}\in D} |[s,t]|$$

if $L = 3$.

Clearly the number of variables and the number of constraints are both polynomial. Hence, the linear relaxation of Formulation (5.15) can be solved in polynomial time using a linear programming method.

5.2.4 Path-Arc formulation

The kHNDP can also be formulated using a path-based model. Every solution of the problem is represented by a collection of directed st-paths in graphs $\widetilde{G}_{st}$, $\{s,t\} \in D$.

Let $\{s,t\} \in D$ and $\mathcal{P}_{st}$ be the set of st-dipaths in $\widetilde{G}_{st}$. Given a directed path $\widetilde{P} \in \mathcal{P}_{st}$, we denote by $\Gamma^{st}_{\widetilde{P}} = (\gamma^{st}_{\widetilde{P},a})_{a\in\widetilde{A}_{st}}$ the *incidence vector* of $\widetilde{P}$ that is the vector given by $\gamma^{st}_{\widetilde{P},a} = 1$ if $a \in \widetilde{P}$ and $\gamma^{st}_{\widetilde{P},a} = 0$ otherwise. Given a subgraph H of G and a set of subgraphs $\widetilde{H}_{st}$ of $\widetilde{G}_{st}$, $\{s,t\} \in D$, we let $\mu^{st}_{\widetilde{H}_{st}} \in \mathbb{R}^{\mathcal{P}_{st}}$ be the 0-1 vector such that $\mu^{st}_{\widetilde{H}_{st}}(\widetilde{P}) = 1$ if $\widetilde{P} \in \mathcal{P}_{st}$ is in $\widetilde{H}_{st}$ and $\mu^{st}_{\widetilde{H}_{st}}(\widetilde{P}) = 0$ otherwise.

If H induces a solution of the kHNDP, then x^H and $(\mu^{st}_{\widetilde{H}_{st}})_{\{s,t\}\in D}$ satisfy the following inequalities.

$$\sum_{\widetilde{P}\in\mathcal{P}_{st}} \mu^{st}(\widetilde{P}) \geq k, \tag{5.16}$$

$$\sum_{\widetilde{P}\in\mathcal{P}_{st}} \gamma^{st}_{\widetilde{P},a}\mu^{st}(\widetilde{P}) \leq x(e), \text{ for all } a \in \widetilde{A}_{st}(e),\ \{s,t\} \in D,\ e \in E, \tag{5.17}$$

$$x(e) \leq 1, \text{ for all edge } e \in E, \tag{5.18}$$

$$\mu^{st}(\widetilde{P}) \geq 0, \text{ for every } \widetilde{P} \in \mathcal{P}_{st}, \{s,t\} \in D, \tag{5.19}$$

where $\mu^{st} \in \mathbb{R}^{\mathcal{P}_{st}}$ and $x \in \mathbb{R}^E$.

Inequalities (5.16) express the fact that the subgraph $\widetilde{G}_{st}$ must contain at least k st-dipaths. Inequalities (5.17) indicate that the st-dipaths are arc-disjoint.

The following theorem gives an integer programming formulation for the kHNDP using the path-based model described above.

Theorem 5.2.3 *The kHNDP for $L = 2, 3$ is equivalent to the following inter program*

$$min\{cx;\ subject\ to\ (5.17) - (5.19),\ x \in \mathbb{Z}_+^E, \mu^{st} \in \mathbb{Z}_+^{\mathcal{P}_{st}},\ for\ all\ \{s,t\} \in D\}. \tag{5.20}$$

Formulation (5.20) is called *Path-Arc formulation* and denoted by kHNDP$_{PA}$. Remark that this formulation contains an exponential number of variables while the number of non trivial inequalities is

$$\sum_{\{s,t\}\in D} |\delta(s)| + \sum_{\{s,t\}\in D} |\delta(t)| - \sum_{\{s,t\}\in D} |[s,t]| - d(n-3)$$

if $L = 2$ and

$$2d|E| - \sum_{\{s,t\}\in D} |\delta(s)| - \sum_{\{s,t\}\in D} |\delta(t)| + \sum_{\{s,t\}\in D} |[s,t]| - d(n-3)$$

if $L = 3$, which is polynomial. To solve the linear relaxation of Formulation (5.20), it is necessary to use appropriate method such as column generation.

In the next section we introduce a further formulation for the kHNDP also based on directed graphs. However, unlike the separated formulations, this formulation is supported by only one directed graph.

5.3 Aggregated formulation for the kHNDP

Let $G = (V, E)$ be an undirected graph, $L \in \{2, 3\}$, $k \geq 2$ be two integers, and D be the demand set. We denote by S_D and T_D respectively the sets of source and destination nodes of D. In the case where two demands $\{s_1, t_1\}$ and $\{s_2, t_2\}$ are such that $s_1 = t_2 = s$, we keep a copy of s in both S_D and T_D.

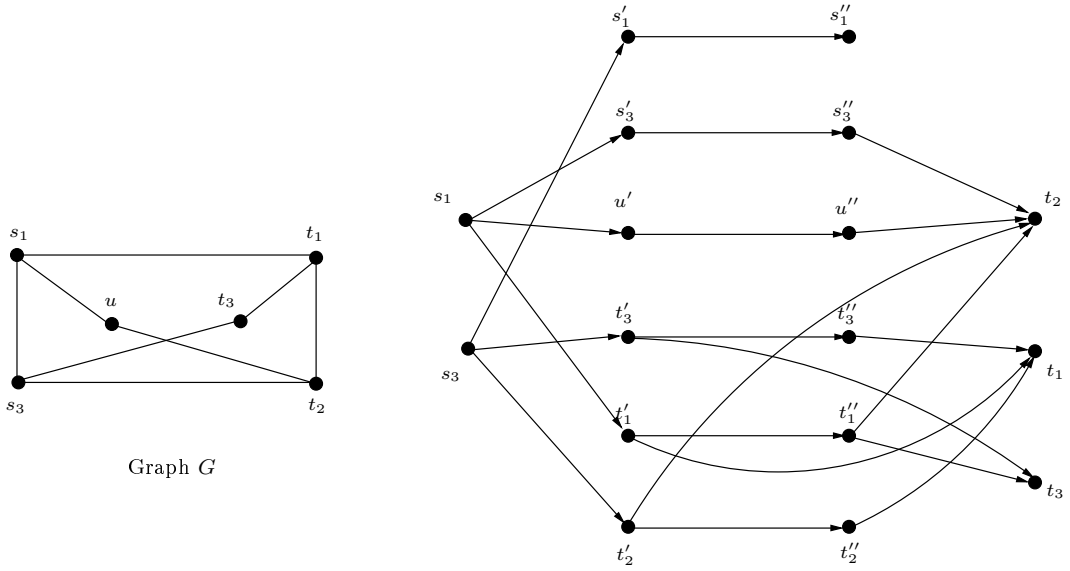
In this section, we will introduce a new formulation for the kHNDP which is supported by a directed graph $\widetilde{G} = (\widetilde{V}, \widetilde{A})$ obtained from G as follows. Let N' and N'' be two copies of V. We denote by u' and u'' the nodes of N' and N'' corresponding to a node $u \in V$. Let $\widetilde{V} = S_D \cup N' \cup N'' \cup T_D$. For every node $u \in V$, we add in $\widetilde{G}$ an arc (u', u''). For each $\{s, t\} \in D$, with $s \in S_D$ and $t \in T_D$, we apply the following procedure.

i) For an edge $e = st$, we add in $\widetilde{G}$ an arc (s, t') and an arc (t', t);

ii) For an edge $su \in E$, $u \in V \setminus \{s, t\}$, we add an arc (s, u') in $\widetilde{G}$;

iii) For an edge $vt \in E$, $v \in V \setminus \{s, t\}$, we add an arc (v'', t).

If $L = 3$, for each edge $e = uv \in E$, we also add two arcs (u', v'') and (v', u'') (see Figures 5.2 and 5.3 for examples with $L = 2$ and $L = 3$).



Graph G

Graph $\widetilde{G}$

Figure 5.2: Construction of graph $\widetilde{G}$ with $D = \{\{s_1, t_1\}, \{s_1, t_2\}, \{s_3, t_3\}\}$ and $L = 2$.

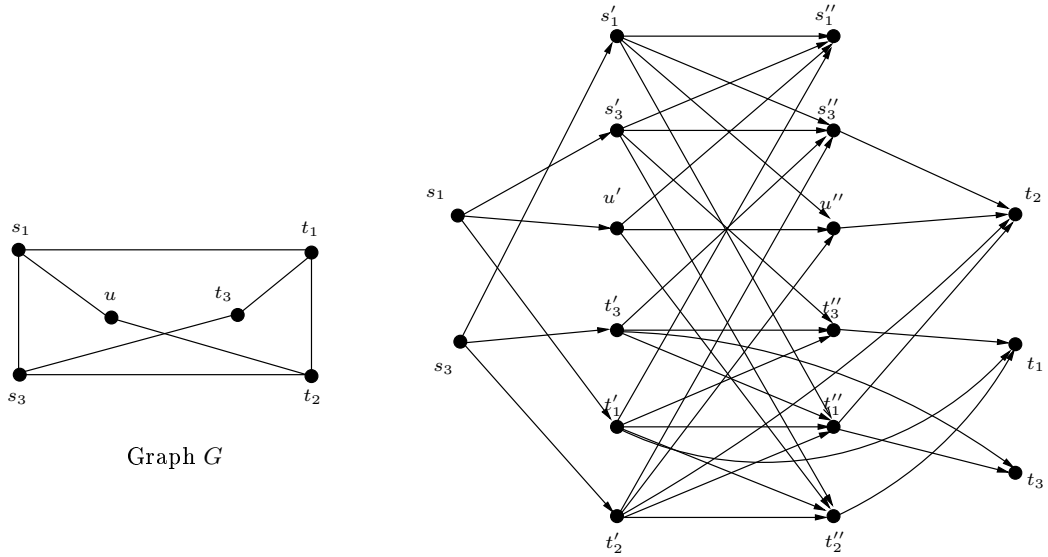


Figure 5.3: Construction of graph $\widetilde{G}$ with $D = \{\{s_1, t_1\}, \{s_1, t_2\}, \{s_3, t_3\}\}$ and $L = 3$.

$\widetilde{G}$ contains $|\widetilde{V}| = 2|V| + |S| + |T|$ nodes and $|\widetilde{A}| = |V| + \sum_{s \in S} |\delta(s)| + \sum_{t \in T} |\delta(t)|$ arcs if $L = 2$ and $|\widetilde{A}| = 2|E| + |V| + \sum_{s \in S} |\delta(s)| + \sum_{t \in T} |\delta(t)|$ arcs if $L = 3$.

If $\widetilde{G} = (\widetilde{V}, \widetilde{A})$ is the graph associated with G, then for an edge $e \in E$, we denote by $\widetilde{A}(e)$ the set of arcs of $\widetilde{G}$ corresponding to e.

Observe that $\widetilde{G}$ is circuitless. Also note that for a given demand $\{s, t\} \in D$, every st-dipath in $\widetilde{G}$ contains at most 3 arcs. An L-st-path $P = (s, u, v, t)$ of G, where u and v may be the same, corresponds to an st-dipath $\widetilde{P} = (s, u', v'', t)$ in $\widetilde{G}$. Conversely, every st-dipath $\widetilde{P} = (s, u', v'', t)$ of $\widetilde{G}$, where u' and v'' may correspond to the same node of V, correponds to an L-st-path $P = (s, u, v, t)$, where u and v may be the same. Moreover $\widetilde{G}$ does not contain any arc of the form (s, s') and (t'', t), for every $s \in S_D$ and $t \in T_D$. If a node $t \in T_D$ appears in exactly one demand $\{s, t\}$, then $[s'', t] = \emptyset$. In the remain of this chapter we will suppose w.l.o.g. that each node of T_D is involved, as destination, in only one demand. In fact, in general, if a node $t \in T_D$ is involved, as destination, in more than one demand, say $\{s_1, t\}, ..., \{s_p, t\}$, with $p \geq 2$, then one may replace in T_D t by p nodes $t_1, ..., t_p$ and in D each demand $\{s_i, t\}$ by $\{s_i, t_i\}$, $i = 1, ..., p$.

We have the following result.

Lemma 5.3.1 *Let $L \in \{2,3\}$. If each node $t \in T_D$ appears in exacly one demand, then for every $\{s,t\} \in D$,*

i) if two L-st-paths of G are edge-disjoint, then the corresponding st-dipaths of $\widetilde{G}$ are arc-disjoint.

ii) if two st-dipaths of $\widetilde{G}$ are arc-disjoint, then the corresponding st-paths in G contain two edge-disjoint L-st-paths.

Proof. The proof will be given for $L = 3$. It follows the same lines for $L = 2$.
i) Let $\{s,t\} \in D$ and let P_1 and P_2 be two edge-disjoint 3-st-paths and $\widetilde{P}_1$ and $\widetilde{P}_2$ be the two st-dipaths of $\widetilde{G}$ corresponding to P_1 and P_2. We will show that $\widetilde{P}_1$ and $\widetilde{P}_2$ are arc-disjoint. Suppose the contrary that is $\widetilde{P}_1$ and $\widetilde{P}_2$ intersect on an arc $a \in \widetilde{A}$ of the form either (s,t'), (s,u'), (v'',t), (u',v'') or (u',u''), with $u' \in N'$ and $v'' \in N''$. If a is of the form (s,t'), (s,u'), (v'',t) or (u',v''), then it corresponds to an edge e of G of the form either st, su, vt or uv. It then follows that P_1 and P_2 both contain edge e, a contradiction. If $\widetilde{P}_1$ and $\widetilde{P}_2$ intersect on an arc of the form (u',u''), then they also contain arcs of the form (s,u') and (u'',t). Thus, P_1 and P_2 also contain simultaneously the edges su and ut, a contradiction. Thus, $\widetilde{P}_1$ and $\widetilde{P}_2$ are arc-disjoint.

ii) Let $\widetilde{P}_1$ and $\widetilde{P}_2$ be two arc-disjoint st-dipaths of $\widetilde{G}$ and suppose that P_1 and P_2, the 3-st-paths of G corresponding to $\widetilde{P}_1$ and $\widetilde{P}_2$, are not edge-disjoint. Thus P_1 and P_2 intersect on edges of the form either st, su, vt or uv, with $u, v \neq s, t$.

If P_1 and P_2 intersect edge st, then each path $\widetilde{P}_1$ and $\widetilde{P}_2$ contains at least one arc among those corresponding to st in $\widetilde{G}$, that is (s,t'), (s',t'') or (t',s''). If $\widetilde{P}_1$ and $\widetilde{P}_2$ contain (s',t''), then they should also contain arc (s,s'). Since $[s,s'] = \emptyset$, this is impossible. In a similar way, we show that $\widetilde{P}_1$ and $\widetilde{P}_2$ cannot contain (t',s''). Hence, $\widetilde{P}_1$ and $\widetilde{P}_2$ both contain arc (s,t'), a contradiction.

If P_1 and P_2 intersect on su, then each path $\widetilde{P}_1$ and $\widetilde{P}_2$ contains either (s,u'), (s',u'') or (u',s''). Since $[s,s'] = \emptyset = [s'',t]$, $\widetilde{P}_1$ and $\widetilde{P}_2$ should both use arc (s,u'), a contradiction.

If P_1 and P_2 intersect on vt, then $\widetilde{P}_1$ and $\widetilde{P}_2$ contain either (v',t''), (t',v'') or (v'',t). As $[t'',t] = \emptyset$, $\widetilde{P}_1$ and $\widetilde{P}_2$ cannot use arc (v',t''). Moreover, if $\widetilde{P}_1$ or $\widetilde{P}_2$ contains (t',v''), then it also contains arc (v'',t). Hence, $\widetilde{P}_1$ and $\widetilde{P}_2$ both contain arc (v'',t), a contradiction.

In consequence, $P_1 \cap P_2 = \{uv\}$, $u, v \neq s, t$. This implies that $\widetilde{P}_1$ and $\widetilde{P}_2$ are respectively of the form $(su', u'v'', v''t)$ and $(sv', v'u'', u''t)$, and $P_1 = (su, uv, vt)$ and $P_2 = (su, vu, ut)$. Let $P_1' = (su, ut)$ and $P_2' = (sv, vt)$. Clearly P_1' and P_2' are edge-disjoint. Since they are of length 2, we simply associate $\widetilde{P}_1$ and $\widetilde{P}_2$ with them, which

ends the proof of the lemma. □

As a consequence of Lemma 5.3.1, the graph G contains k edge-disjoint L-st-paths for a demand $\{s,t\}$ if and only if $\widetilde{G}$ contains at least k arc-disjoint st-dipaths. Thus we have the following corrollary.

Corollary 5.3.1 *Let H be a subgraph of G and $\widetilde{H}$ the subgraph of $\widetilde{G}$ obtained by considering all the arcs of $\widetilde{G}$ corresponding to the edges of H toghether with the arcs of the form (u',u''), $u \in V$, and (t',t), for every $t \in T_D$. Then H induces a solution of the kHNDP if $\widetilde{H}$ is a solution of the Survivable Directed Network Design Problem (kDNDP). Conversely, if $\widetilde{H}$ is a subgraph of $\widetilde{G}$ and H is the subgraph of G obtained by considering all the edges which correspond to at least one arc of $\widetilde{H}$, then H induces a solution of the kHNDP only if $\widetilde{H}$ is a solution of the kDNDP.*

By Menger's Theorem, $\widetilde{G}$ contains k arc-disjoint st-dipaths if and only if every st-dicut of $\widetilde{G}$ contains at least k arcs. Let $x \in \mathbb{R}^E$ and $y \in \mathbb{R}^{\widetilde{A}}$. If $\widetilde{H}$ is a solution of the kDNDP and H is the subgraph of G whose edges correspond to the arcs of $\widetilde{H}$, then x^H and $y^{\widetilde{H}}$, the incidence vectors of H and $\widetilde{H}$, satisfy the following inequalities

$$y(\delta^+(\widetilde{W})) \geq k, \quad \text{for all } st\text{-dicut } \delta^+(\widetilde{W}), \{s,t\} \in D, \tag{5.21}$$

$$y(a) \leq x(e), \qquad \text{for all } a \in \widetilde{A}(e),\ e \in E, \tag{5.22}$$

$$y(a) \geq 0, \qquad \text{for all } a \in \widetilde{A}, \tag{5.23}$$

$$x(e) \leq 1, \qquad \text{for all } e \in E. \tag{5.24}$$

We have the following theorem, which easily follows from Corollary 5.3.1.

Theorem 5.3.1 *The kHNDP for $L = 2,3$ is equivalent to the following integer program*

$$min\{cx;\ \textit{subject to } (5.21) - (5.24),\ x \in \mathbb{Z}_+^E, y \in \mathbb{Z}_+^{\widetilde{A}}\}. \tag{5.25}$$

Formulation (5.25) will be called *Aggregated formulation* and denoted by $k\text{HNDP}_{Ag}$. Inequalities (5.21) will be called *directed st-cut inequalities* or *st-dicut inequalities* and (5.22) will be called *linking inequalities*. The latter inequalities indicate that an arc a, corresponding to an edge e, is not in $\widetilde{H}$ if e is not taken in H. Inequalities (5.23) and (5.24) are called *trivial inequalities*.

This formulation contains $|E| + |\widetilde{A}| = |E| + |V| + \sum_{s \in S_D} |\delta(s)| + \sum_{t \in T_D} |\delta(t)|$ variables if $L = 2$ and $|E| + |\widetilde{A}| = 3|E| + |V| + \sum_{s \in S_D} |\delta(s)| + \sum_{t \in T_D} |\delta(t)|$ variables if $L = 3$. The number of constraints is exponential since the st-dicuts are in exponential number. But, as it will turn out, the separation problem of inequalities (5.21) can be solved in polynomial time and hence, the linear relaxation of (5.25) so is.

In the next section, we present a comparitive study of different formulations presented in the last section. In particular, we will show that the values of the linear relaxations of the separated and Aggregated formulations are greater than that of the Natural formulation and thus, these formulations are as strong as the Natural formulation.

5.4 Separated and Aggregated formulations versus Natural formulation

Here we show that the values of the linear relaxations of Formulations (5.10)-(5.25), are greater than that of the Natural formulation of the kHNDP. For this, we show that a solution $\overline{x}$ of the linear relaxation of any of these four formulations is also a solution of the linear relaxation of Formulation (5.5).

5.4.1 Separated formulations versus Natural formulation

We first consider the Cut, Node-Arc and Path-Arc formulations. We will examine the Node-Arc formulation, the proof for the Cut and Path-Arc formulations is along the same lines. We will show that, if a vector $\overline{x} \in \mathbb{R}^E$ and $|D|$ flow vectors $\overline{f}^{st} \in \mathbb{R}^{\widetilde{A}_{st}}$, $\{s,t\} \in D$, induce a solution of the linear relaxation of (5.15), then $\overline{x}$ also satisfies inequalities (5.1)-(5.4). To this end, we first associate with each digraph $\widetilde{G}_{st}$ a solution $\overline{y}_{st} \in \mathbb{R}^{\widetilde{A}_{st}}$ obtained from $\overline{x}$. Then we introduce a procedure which permits to associate with every st-cut and L-st-path-cut of G an st-dicut of $\widetilde{G}_{st}$ with the same value regarding $\overline{y}_{st}$.

For all $\{s,t\} \in D$, let $\overline{y}_{st} \in \mathbb{R}^{\widetilde{A}_{st}}$ be the vector given by

$$\overline{y}_{st}(a) = \begin{cases} \overline{x}(su) & \text{if } a \text{ is of the form } (s,u),\ u \in N_{st}, \\ \overline{x}(vt) & \text{if } a \text{ is of the form } (v',t),\ v' \in N'_{st}, \\ \overline{x}(uv) & \text{if } a \text{ is of the form } (u,v') \text{ or } (v',u), \\ & u,v \in N_{st},\ u',v' \in N'_{st},\ u \neq v,\ u' \neq v', \\ \overline{x}(st) & \text{if } a \text{ is of the form } (s,t), \\ 1 & \text{if } a \text{ is of the form } (u,u'),\ u \in N_{st},\ u' \in N'_{st}. \end{cases}$$

Note that, since $\overline{f}^{st}$ is of value $\geq k$, for all $\{s,t\} \in D$, by inequalities (5.12), it follows that $\overline{y}_{st}(\delta^+(\widetilde{W})) \geq k$ for all st-dicut $\delta^+(\widetilde{W})$ of $\widetilde{G}_{st}$.

Now we introduce a procedure, called *Procedure A*, which, for a demand $\{s,t\}$ and an edge set $C \subseteq E$, produces an arc subset $\widetilde{C}$ of $\widetilde{G}_{st}$.

i) For an edge $st \in C$, add the arc (s,t) in $\widetilde{C}$;

ii) for an edge $su \in C$, add the arc (s,u) in $\widetilde{C}$, $u \in N_{st}$;

iii) for an edge $vt \in C$, add the arc (v',t) in $\widetilde{C}$, $v' \in N'_{st}$;

iv) for an edge $uv \in C$, $u \neq v$, $u,v \in V \setminus \{s,t\}$,

iv.1) if $su \in C$ or $vt \in C$, then add (v,u') in $\widetilde{C}$, with $v \in N_{st}$ and $u' \in N'_{st}$;

iv.2) if $su \notin C$ and $vt \notin C$, then add the arc (u,v') in $\widetilde{C}$.

Observe that $\widetilde{C}$ does not contain any arc of the form (u,u') with $u \in N_{st}$ and $u' \in N'_{st}$. Also note that $\widetilde{C}$ does not contain at the same time two arcs (u,v') and (v,u'), for an edge $uv \in E$ with $u,v \in V \setminus \{s,t\}$.

Conversely, an arc subset $\widetilde{C}$ of $\widetilde{A}_{st}$ can be obtained from an edge set $C \subseteq E$, using Procedure A, if $\widetilde{C}$ does not contain simultaneously two arcs (u,v') and (v,u'), $u,v \in N_{st}$, $u',v' \in N'_{st}$, and does not contain any arc of the form (u,u') with $u \in N_{st}$, $u' \in N'_{st}$.

As each arc of C corresponds to a single arc of $\widetilde{C}$ and vice versa, C and $\widetilde{C}$ have the same weight with respect to $\overline{x}$ and $\overline{y}$, that is $\overline{x}(C) = \overline{y}_{st}(\widetilde{C})$.

Lemma 5.4.1 *Let $(\overline{x}, \overline{f}^{s_1t_1}, ..., \overline{f}^{s_dt_d})$ be a solution of the linear relaxation of Formulation (5.15). Let $C \subseteq E$ be an edge set of G which is an st-cut or a L-st-path-cut induced by a partition $(V_0, ..., V_{L+1})$ such that $|V_0| = |V_{L+1}| = 1$, with $L \in \{2,3\}$. Also let $\overline{y}_{st} \in \mathbb{R}^{\widetilde{A}_{st}}$ be the solution obtained from $\overline{x}$ and $\widetilde{G}_{st}$. Then the arc set $\widetilde{C}$ obtained from C by Procedure A is an st-dicut of $\widetilde{G}_{st}$. Moreover, $\overline{x}(C) = \overline{y}_{st}(\widetilde{C})$.*

Proof. Similar to that of Lemma 4.3.1. □

By Lemma 5.4.1, every st-cut and L-st-path-cut C of G, induced by a partition $(V_0, ..., V_{L+1})$ such that $|V_0| = |V_{L+1}| = 1$, corresponds to an st-dicut $\widetilde{C}$ of $\widetilde{G}_{st}$ of the same weight, that is $\overline{x}(C) = \overline{y}_{st}(\widetilde{C})$. As by the remark above, $\overline{y}_{st}(\widetilde{C}) \geq k$, for every st-dicut of $\widetilde{G}_{st}$, we have that $\overline{x}(C) \geq k$. Therefore, $\overline{x}$ satisfies inequalities (5.1)-(5.4).

This result implies that if a vector $\overline{x}$ and a set of flow vectors $(\overline{f}^{st})_{\{s,t\}\in D}$ inducing an optimal solution of the linear relaxation of Formulation (5.15), then $\overline{x}$ is a solution of the linear relaxation of (5.5). This yields the theorem below.

Theorem 5.4.1 *If Z^*_{NA} (resp. Z^*_{Cut}) (resp. Z^*_{PA}) is the value of the linear relaxation of Formulation (5.15) (resp. (5.10)) (resp. (5.20)) and Z^*_{nat} is that of Formulation (5.5), then $Z^*_{nat} \leq Z^*_{NA}$ (resp. $Z^*_{nat} \leq Z^*_{Cut}$) (resp. $Z^*_{nat} \leq Z^*_{PA}$).*

In the next section we show that this result also holds for the Aggregated formulation.

5.4.2 The linear relaxation of the Aggregated formulation

Consider the Aggregated formulation (5.25) and let $\widetilde{G} = (\widetilde{V}, \widetilde{A})$ be the directed graph associated with G. Let also $(\overline{x}, \overline{y}) \in \mathbb{R}^E \times \mathbb{R}^{\widetilde{A}}$ be a pair of vectors which induces a solution of the linear relaxation of Formulation (5.25). As for the Node-Arc formulation, we are going to associate with every edge set $C \subseteq E$ and demand $\{s,t\} \in D$, an arc set $\widetilde{C}$ of $\widetilde{G}$, and show that if C is an st-cut or an L-st-path-cut induced by a partition $(V_0, ..., V_{L+1})$ with $|V_0| = |V_{L+1}| = 1$, then $\widetilde{C}$ is an st-dicut of $\widetilde{G}$.

For this, we give the following procedure called *Procedure B*. Let $C \subseteq E$ and $\{s,t\} \in D$, and let $\widetilde{C}$ be the arc set of $\widetilde{G}$ obtained as follows.

i) For an edge $st \in C$, add the arc (s,t') in $\widetilde{C}$;

ii) for an edge $su \in C$, add the arc (s,u') in $\widetilde{C}$, $u' \in N'$;

iii) for an edge $vt \in C$, add the arc (v'',t) in $\widetilde{C}$, $v'' \in N''$;

iv) for an edge $uv \in C$, $u \neq v$, $u, v \in V \setminus \{s,t\}$,

iv.1) if $su \in C$ or $vt \in C$, then add (v',u'') in $\widetilde{C}$, with $v' \in N'$ and $u'' \in N''$;

iv.2) if $su \notin C$ and $vt \notin C$, then add the arc (u', v'') in $\widetilde{C}$.

Observe that $\widetilde{C}$ does not contain any arc neither of the form (u', u'') with $u' \in N'$ and $u'' \in N''$, nor of the form (t', t) for $t \in T_D$. Also note that $\widetilde{C}$ does not contain at the same time two arcs (u', v'') and (v', u''), for an edge $uv \in E$.

Conversely, an arc subset $\widetilde{C}$ of $\widetilde{A}$ can be obtained by Procedure B from an edge set $C \subseteq E$ if $\widetilde{C}$ does not contain simultaneously two arcs (u', v'') and (v', u''), $u', v' \in N'$, $u'', v'' \in N''$, and any arc of the form (u', u'') with $u' \in N'$, $u'' \in N''$ and (t', t), $t \in T_D$.

As each arc of C corresponds to an arc of $\widetilde{C}$ and vice versa, and $(\overline{x}, \overline{y})$ satisfies inequalities (5.22), we have that $\overline{x}(C) \geq \overline{y}(\widetilde{C})$. We then have the following result given without proof since its proof is similar to that of Lemma 4.3.1.

Lemma 5.4.2 *Let $(\overline{x}, \overline{y})$ be a solution of the linear relaxation of Formulation (5.25). Let $C \subseteq E$ be an edge set of G which is an st-cut or a L-st-path-cut induced by a partition $(V_0, ..., V_{L+1})$ such that $|V_0| = |V_{L+1}| = 1$, with $L \in \{2, 3\}$. Then the arc set obtained from C and $\{s, t\}$ by Procedure B is an st-dicut of $\widetilde{G}$. Moreover, $\overline{x}(C) \geq \overline{y}(\widetilde{C})$.*

Proof. The proof is similar to that of Lemma 4.3.1. □

By Lemma 5.4.2, every st-cut and L-st-path-cut C of G corresponds to an st-dicut $\widetilde{C}$ of $\widetilde{G}$ such that $\overline{x}(C) \geq \overline{y}(\widetilde{C})$. As $(\overline{x}, \overline{y})$, induces a solution of the linear relaxation of Formulation (5.25), and hence, $\overline{y}(\widetilde{C}) \geq k$, for every st-dicut $\widetilde{C}$ of $\widetilde{G}$, we have that $\overline{x}(C) \geq k$. Therefore, $\overline{x}$ satisfies inequalities (5.1)-(5.4), yielding the theorem below.

Theorem 5.4.2 *If Z^*_{Ag} is the optimal solution of Formulation (5.25) and Z^*_{nat} is the optimal solution of Formulation (5.5), then $Z^*_{nat} \leq Z^*_{Ag}$.*

The next section is devoted to a polyhedral study of the different formulations introduced before. For the polytope associated with each formulation we describe some classes of valid inequalities and give some conditions under which these inequalities define facets.

5.5 The kHNDP polytopes

Let $G = (V, E)$ be an undirected graph, $L \in \{2, 3\}$ and $k \geq 2$ two integers, and $D = \{\{s_1, t_1\}, ..., \{s_d, t_d\}\}$, $d \geq 2$, the set of demands.

We will denote by $k\text{HNDP}_{Ag}(G, D)$ (resp. $k\text{HNDP}_{Cu}(G, D)$) (resp. $k\text{HNDP}_{NA}(G, D)$) (resp. $k\text{HNDP}_{PA}(G, D)$) the polytope associated with the Aggregated formulation (resp. Cut formulation), (resp. Node-Arc formulation), (resp. Path-Arc formulation).

5.5.1 The polytope $k\text{HNDP}_{Ag}(G, D)$

We first consider the polytope $k\text{HNDP}_{Ag}(G, D)$. Let $\widetilde{G} = (\widetilde{V}, \widetilde{A})$ be the directed graph associated with G and D in the case of the Aggregated formulation. Let E^* be the set of edges $e \in E$ such that there exists a demand $\{s, t\} \in D$ such that $G \setminus \{e\}$ does not contain k edge-disjoint L-st-paths. Such an edge is said to be L-st-*essential*. Also consider an arc $a \in \widetilde{A}$ such that there exists a demand $\{s, t\} \in D$ such that the graph $\widetilde{G} \setminus \{a\}$ does not contain k arc-disjoint st-dipaths. Such an arc a is said to be st-*essential*. We will denote by $\widetilde{A}^*$ the set of st-essential arcs of $\widetilde{G}$.

The following theorem characterizes the dimension of $k\text{HNDP}_{Ag}(G, D)$.

Theorem 5.5.1 $dim(kHNDP_{Ag}(G, D)) = |E| + |\widetilde{A}| - |E^*| - |\widetilde{A}^*|$.

Proof. Obviously, we have that $dim(k\text{HNDP}_{Ag}(G, D)) \leq |E| + |\widetilde{A}| - |E^*| - |\widetilde{A}^*|$. Now we show that $dim(k\text{HNDP}_{Ag}(G, D)) \geq |E| + |\widetilde{A}| - |E^*| - |\widetilde{A}^*|$. For this, we show that the maximum number of affinely independant solutions of $k\text{HNDP}_{Ag}(G, D)$ is greater than or equal to $|E| + |\widetilde{A}| - |E^*| - |\widetilde{A}^*| + 1$. Recall that a solution of $k\text{HNDP}_{Ag}(G, D)$ is described by a pair $(\widetilde{F}, F)$ where $\widetilde{F} \subseteq \widetilde{A}$ and $F \subseteq E$ is the associated edge set. Also note that an edge set F induces a solution of the kHNDP if and only if the associated arc set $\widetilde{F}$ induces a subgraph of $\widetilde{G}$ containing k arc-disjoint st-dipaths for every $\{s, t\} \in D$.

Consider the pairs $(\widetilde{A} \setminus \{a\}, E)$, for all $a \in \widetilde{A} \setminus \widetilde{A}^*$. As $a \notin \widetilde{A}^*$, these pairs induce solutions of $k\text{HNDP}_{Ag}(G, D)$.

For every edge $e \in E \setminus E^*$, consider the pair $(\widetilde{A} \setminus \widetilde{A}(e), E \setminus \{e\})$. Remind that, for all $e \in E$, $\widetilde{A}(e)$ is the set of arcs of $\widetilde{A}$ corresponding to e. As $e \in E \setminus E^*$, the subgraph induced by $E \setminus \{e\}$ contains k edge-disjoint L-st-paths for every $\{s, t\} \in D$

and the subgraph of $\widetilde{G}$ induced $\widetilde{A} \setminus \widetilde{A}(e)$ also contains k arc-disjoint st-dipaths for every $\{s,t\} \in D$. Hence this pair induces a solution of $k\text{HNDP}_{Ag}(G,D)$.

One can easily observe that these solutions, toghether with the solution given by the pair $(\widetilde{A}, E)$, form a family of $|E \setminus E^*| + |\widetilde{A} \setminus \widetilde{A}^*| + 1$ solutions of the $k\text{HNDP}_{Ag}$ that are affinely independant. Therefore, $dim(k\text{HNDP}_{Ag}(G,D)) \geq |E| + |\widetilde{A}| - |E^*| - |\widetilde{A}^*|$, which ends the proof of the theorem. □

Consequently, $k\text{HNDP}_{Ag}(G,D)$ is full dimensional if and only if $E^* = \emptyset = \widetilde{A}^*$.

The next theorem shows that if G is complete and $|V| \geq k+2$, then $E^* = \emptyset = \widetilde{A}^*$, implying that $k\text{HNDP}_{Ag}(G,D)$ is full dimensional. But before, we give the following lemma.

Lemma 5.5.1 *If G is complete, then for every $\{s,t\} \in D$, there exist at least $|V|-1$ arc-disjoint st-dipaths in $\widetilde{G}$.*

Proof. Suppose that G is complete. Consider a demand $\{s,t\} \in D$ and the arc set $\widetilde{H} = [s,N'] \cup [N',N''] \cup [N'',t] \cup [t',t]$. Clearly, since G is complete, $|[s,N']| = |V|-1$, $|[N'',t]| = |V|-2$. Moreover, by the construction of $\widetilde{G}$, $|[N',N'']| = |V|$ and $|[t',t]| \geq 1$. Thus, the subgraph induced by $\widetilde{H}$ contains $|V|-1$ arc-disjoint st-dipaths in $\widetilde{G}$. □

A consequence of Lemma 5.5.1 is that for a complete graph G with $|V| \geq k+2$, the graph $\widetilde{G}$ contains at least $k+1$ arc-disjoint st-dipaths for every $\{s,t\} \in D$. This implies that $E^* = \emptyset = \widetilde{A}^*$. We thus have the following.

Corollary 5.5.1 *If G is complete and $|V| \geq k+2$, then $kHNDP_{Ag}(G,D)$ is full dimensional.*

In what follows, we give necessary and sufficient conditions for the trivial inequalities to define facets of $k\text{HNDP}_{Ag}(G,D)$. Remark that the inequalities $y(a) \leq 1$, for all $a \in \widetilde{A}$, and $x(e) \geq 0$, for all $e \in E$, are redundant with respect to the inequalities

$$\begin{aligned} y(a) &\geq 0 \qquad \text{for all } a \in \widetilde{A}, \\ x(e) &\leq 1 \qquad \text{for all } e \in E, \\ y(a) &\leq x(e) \quad \text{for all arc } a \in \widetilde{A}(e), \end{aligned}$$

and hence, do not define facets.

Theorem 5.5.2 *If G is complete and $|V| \geq k+2$, then the following hold.*

i) Every inequality $x(e) \leq 1$ defines a facet of $kHNDP_{Ag}(G, D)$;

ii) An inequality $y(a) \geq 0$ defines a facet of $kHNDP_{Ag}(G, D)$ if and only either $|V| \geq k+3$ or $|V| = k+2$ and a does not belong to an st-dicut of $\widetilde{G}$ of cardinality $k+1$.

Proof. First note that, as G is complete and $|V| \geq k+2$, by Corollary 5.5.1, $k\text{HNDP}_{Ag}(G, D)$ is full dimensional.

i) Let $a \in \widetilde{A}$. Since G is complete and $|V| \geq k+2$, the subgraph induced by $\widetilde{A} \setminus \{a\}$ contains k arc-disjoint st-dipaths for every $\{s, t\} \in D$. Thus, the pair $(\widetilde{A} \setminus \{a\}, E)$ induces a solution of $k\text{HNDP}_{Ag}(G, D)$. Moreover, its incidence vector satisfies $x(e) = 1$.

Now let $f \in E \setminus \{e\}$. As before, the subgraph induced by $E \setminus \{f\}$ contains k edge-disjoint L-st-paths, for every $\{s, t\} \in D$. Thus, the pair $(\widetilde{A} \setminus \widetilde{A}(f), E \setminus \{f\})$ induces a solution of $k\text{HNDP}_{Ag}(G, D)$, whose incidence vector satisfies $x(e) = 1$. Recall that $\widetilde{A}(f)$ denotes the set of arcs of $\widetilde{G}$ corresponding to f.

It is not hard to see that these two families of solutions, toghether with the solution induced by the pair $(\widetilde{A}, E)$, form $|E| + |\widetilde{A}|$ solutions whose incidence vectors satisfy $x(e) = 1$ and are affinely independant. This yields $x(e) \leq 1$ defines a facet of $k\text{HNDP}_{Ag}(G, D)$.

ii) Consider an arc $a \in \widetilde{A}$ and suppose that $|V| \geq k+3$. By Lemma 5.5.1, $\widetilde{G}$ contains at least $k+2$ arc-disjoint st-dipaths for every $\{s, t\} \in D$, and G contains at least $k+2$ edge-disjoint L-st-paths. Thus for an edge $e \in E$, the pair $(\widetilde{A} \setminus (\{a\} \cup \widetilde{A}(e)), E \setminus \{e\})$ induces a solution of $k\text{HNDP}_{Ag}(G, D)$. Also, for an arc $a' \in \widetilde{A} \setminus \{a\}$, the pair $(\widetilde{A} \setminus \{a, a'\}, E)$ induces a solution of $k\text{HNDP}_{Ag}(G, D)$. These solution toghether with the solution $(\widetilde{A} \setminus \{a\}, E)$ form a family of $|\widetilde{A}| + |E|$ solutions whose incidence vectors satisfy $y(a) = 0$ and are affinely independant. Thus, $y(a) \geq 0$ defines a facet.

Now suppose that $|V| = k+2$. If a belongs to an st-dicut $\delta^+(\widetilde{W})$ of $k+1$ arcs, then $y(a) \geq 0$ is redundant with respect to the inequalities

$$\begin{aligned} & y(\delta^+(\widetilde{W})) \geq k, \\ & -y(a') \geq -1, \text{ for every arc } a' \in \delta^+(\widetilde{W}) \setminus \{a\}, \end{aligned}$$

and hence cannot define a facet. If a does not belong to an st-dicut of $k+1$ arcs, then, the pairs $(\widetilde{A} \setminus (\{a\} \cup \widetilde{A}(e)), E \setminus \{e\})$, for all $e \in E$, and $(\widetilde{A} \setminus \{a, a'\}, E)$, for

all $a' \in \widetilde{A} \setminus \{a\}$ induce solutions of $k\text{HNDP}_{Ag}(G, D)$. These solutions toghether with the solution $(\widetilde{A} \setminus \{a\}, E)$ form a family of $|\widetilde{A}| + |E|$ solutions whose incidence vectors satisfy $y(a) = 0$ and are affinely independant. Thus $y(a) \geq 0$ defines a facet of $k\text{HNDP}_{Ag}(G, D)$. □

The next theorem gives necessary and sufficient conditions for the directed st-cut inequalities to define facets of $k\text{HNDP}_{Ag}(G, D)$.

Theorem 5.5.3 *Suppose that G is complete and $|V| \geq k+2$ and let $\widetilde{W} \subseteq \widetilde{V}$ be a node set such that there is a demand $\{s, t\} \in D$ with $s \in S_D \cap \widetilde{W}$ and $t \in T_D \cap (\widetilde{V} \setminus \widetilde{W})$ (Recall that S_D (resp. T_D) is the set of terminals of G that are source (resp. destination) in at least one demand). Then the st-dicut inequality $y(\delta^+(\widetilde{W})) \geq k$ defines a facet of $kHNDP_{Ag}(G, D)$ only if the following conditions hold*

i) $\widetilde{W} \cap S_D = \{s\}$ and $(\widetilde{V} \setminus \widetilde{W}) \cap T_D = \{t\}$;

ii) $s' \in \widetilde{V} \setminus \widetilde{W}$, $s'' \in \widetilde{W}$ and $t'' \in \widetilde{W}$.

Proof. We will only show the first condition of i). The proof for ii) follows the same lines. Suppose on the contrary that there exists another node $s_1 \neq s$ in $\widetilde{W} \cap S_D$. Since $s_1 \in S_D$, we have that $[s, s_1] = \emptyset$. Thus, $\delta^+(\widetilde{W} \setminus \{s_1\}) = \delta^+(\widetilde{W}) \setminus \delta^+(s_1)$. Note that the edges of G associated with those of $\delta^+(s_1)$ are those of $\delta(s_1)$. As G is complete, $\delta^+(s_1) \neq \emptyset$. Therefore, the st-dicut inequality induced by $\widetilde{W}$ is redundant with respect to the inequalities

$$\begin{aligned} & y(\delta^+(\widetilde{W} \setminus \{s_1\})) \geq k, \\ & y(a) \geq 0 \qquad\qquad \text{for all } a \in \delta^+(s_1), \end{aligned}$$

and hence, cannot define a facet. □

5.5.2 The polytope $k\text{HNDP}_{Cu}(G, D)$

Now we consider the Cut formulation. The results of this section will be given without proof. In fact their proofs are similar to those of the previous section.

As before, we denote by E^* the set of L-st-essential edges of G and $\widetilde{A}^*_{st}$ the set of st-essential arcs of $\widetilde{G}_{st}$, for every $\{s, t\} \in D$. The following theorem gives the dimension of $k\text{HNDP}_{Cu}(G, D)$.

Theorem 5.5.4 *$dim(kHNDP_{Cu}(G,D)) = |E| + \sum_{\{s,t\}\in D} |\widetilde{A}_{st}| - |E^*| - \sum_{\{s,t\}\in D} |\widetilde{A}^*_{st}|$.*

Proof. Similar to proof of Theorem 5.5.1. □

Lemma 5.5.2 *If G is complete, then for every demand $\{s,t\} \in D$, there exists at least $|V|-1$ arc-disjoint st-dipaths in $\widetilde{G}_{st}$.*

Proof. Similar to proof of Lemma 5.5.1. □

As a consequence, we have the following corollary.

Corollary 5.5.2 *If G is complete and $|V| \geq k+2$, then $kHNDP_{Cu}(G,D)$ is full dimensional.*

Note that the inequalities $y_{st}(a) \leq 1$ and $x(e) \geq 0$ are redundant with respect to $y_{st}(a) \geq 0$, $x(e) \leq 1$ and $y_{st}(a) \leq x(e)$. The next theorem gives necessary and sufficient conditions for inequalities (5.8) and (5.9) to define facets.

Theorem 5.5.5 *If G is complete and $|V| \geq k+2$, then the following hold.*

i) Every inequality $x(e) \leq 1$ defines a facet of $kHNDP_{Cu}(G,D)$.

ii) An inequality $y(a) \geq 0$ defines a facet of $kHNDP_{Cu}(G,D)$ if and only if either $|V| \geq k+3$ or $|V| = k+2$ and a does not belong to an st-cut of cardinality $k+1$.

Proof. Similar to proof of Theorem 5.5.2. □

In the next section, we describe further classes of valid inequalities for the polytopes discussed above. We also give for some of them necessary and sufficient conditions for these inequalities to be facet defining.

5.6 Valid inequalities

Here we describe various classes of inequalities that are valid for the polytopes kHNDP$_{Ag}(G,D)$, kHNDP$_{Cu}(G,D)$, kHNDP$_{NA}(G,D)$ or kHNDP$_{PA}(G,D)$ when $L \in \{2,3\}$. But before, we give the following lemma.

Lemma 5.6.1 *The following inequalities are valid for kHNDP$_{Ag}(G,D)$, kHNDP$_{Cu}(G,D)$, kHNDP$_{NA}(G,D)$, kHNDP$_{PA}(G,D)$:*

$$x(\delta(W)) \geq k, \text{ for every st-cut } \delta(W) \text{ and every } \{s,t\} \in D,$$
$$x(T) \geq k, \text{ for every L-st-path-cut } T \text{ and every } \{s,t\} \in D.$$

Proof. Easy. □

5.6.1 Aggregated cut inequalities

Here we introduce a class of inequalities that are valid for kHNDP$_{Ag}(G,D)$ and kHNDP$_{Cu}(G,D)$. This class of inequalities are inspired from those introduced by Dahl [29] for the polytope of the Survivable Directed Network Design Problem (kDNDP). The kDNDP consists, given a directed graph $\widetilde{H}$, a set of demands D and an integer $k \geq 2$, in finding a minimum weight subgraph of $\widetilde{H}$ which contains k arc-disjoint st-dipaths for every demand $\{s,t\} \in D$. We will first describe these inequalities for kHNDP$_{Ag}(G,D)$ and then extend it to kHNDP$_{Cu}(G,D)$.

5.6.1.1 Aggregated cut inequalities for kHNDP$_{Ag}(G,D)$

Let $\{\widetilde{W_1},...,\widetilde{W_p}\}$, $p \geq 2$, be a family of node sets of $\widetilde{V}$ such that each set $\widetilde{W_i}$ induces an st-dicut of $\widetilde{G}$, for some $\{s,t\} \in D$, and $\widetilde{F_i^0} \subseteq \delta^+_{\widetilde{G}}(\widetilde{W_i})$. Let $\widetilde{F} = \bigcup_{i=1}^{p} [\delta^+_{\widetilde{G}}(\widetilde{W_i}) \setminus \widetilde{F_i^0}]$ and, for an arc $a \in \widetilde{A}$, let $r(a)$ be the number of sets $\delta^+_{\widetilde{G}}(\widetilde{W_i}) \setminus \widetilde{F_i^0}$ which contain the arc a. Note that if $a \in \widetilde{A}$ does not belong to any set $\delta^+_{\widetilde{G}}(\widetilde{W_i}) \setminus \widetilde{F_i^0}$, then $r(a) = 0$. For an edge $e \in E$ and an arc subset $\widetilde{U} \subseteq \widetilde{A}$, we let

$$r'(e,\widetilde{U}) = \sum_{a \in \widetilde{A}(e) \cap \widetilde{U}} r(a), \text{ for all } e \in E.$$

The inequalities below are valid for kHNDP$_{Ag}$

$$\begin{aligned} & y(\delta^+_{\widetilde{G}}(\widetilde{W}_i)) \geq k \text{ for } i = 1, ..., p, \\ & -y(a) \geq -1 \quad \text{for all } a \in \widetilde{F}^0_i,\ i = 1, ..., p. \end{aligned}$$

By summing these inequalities, we obtain

$$\sum_{a \in \widetilde{F}} r(a)y(a) \geq kp - \sum_{i=1}^{p} |\widetilde{F}^0_i|.$$

If $\widetilde{F}_1$ (resp. $\widetilde{F}_2$) denotes the set of arcs $a \in \widetilde{F}$ such that $r(a)$ is odd (resp. even), then the previous inequality can be written as

$$\sum_{a \in \widetilde{F}_1} r(a)y(a) + \sum_{a \in \widetilde{F}_2} r(a)y(a) \geq kp - \sum_{i=1}^{p} |\widetilde{F}^0_i|. \tag{5.26}$$

Let $\widetilde{F}^2_1 \subseteq \widetilde{F}_1$ such that, for every edge $e \in E$ corresponding to an arc of $\widetilde{F}_1$, $r'(e, \widetilde{F}^2_1)$ is even. Let E_2 be the set of edges corresponding to the arcs of $\widetilde{F}^2_1$. By summing inequality (5.26) with the inequalities

$$r(a)x(e) \geq r(a)y(a), \text{ for all } a \in \widetilde{F}^2_1 \text{ and } e \text{ corresponding to } a,$$

we get

$$\sum_{e \in E_2} r'(e, \widetilde{F}^2_1)x(e) + \sum_{a \in \widetilde{F}_1 \setminus \widetilde{F}^2_1} r(a)y(a) + \sum_{a \in \widetilde{F}_2} r(a)y(a) \geq kp - \sum_{i=1}^{p} |\widetilde{F}^0_i|. \tag{5.27}$$

By dividing by 2 and rounding up the right hand side of inequality (5.27), we obtain the following inequality

$$\sum_{e \in E_2} \frac{r'(e, \widetilde{F}^2_1)}{2} x(e) + \sum_{a \in \widetilde{F}_1 \setminus \widetilde{F}^2_1} \frac{r(a)+1}{2} y(a) + \sum_{a \in \widetilde{F}_2} \frac{r(a)}{2} y(a) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}^0_i|}{2} \right\rceil. \tag{5.28}$$

Inequalities of type (5.28) will be called *aggregated cut inequalities*. We give the following result which directly comes from the above description.

Theorem 5.6.1 *Inequalities of type (5.28) are valid for kHNDP$_{Ag}(G,D)$ when $L \in \{2,3\}$.*

Inequalities (5.28) are produced by families of st-dicuts of $\widetilde{G}$ which may have different forms of configurations for the node sets $\widetilde{W}_1,...,\widetilde{W}_p$, $p \geq 2$, and the arc sets $\widetilde{F}_i^0 \subseteq \delta^+_{\widetilde{G}}(\widetilde{W}_i)$, $i = 1,...,p$. In the following, we discuss a special case of these inequalities.

Let $\{\widetilde{W}_1,...,\widetilde{W}_p\}$, $p \geq 2$, be a family of node sets of $\widetilde{V}$ such that each set $\widetilde{W}_i$, $i = 1,...,p$, induces an st-dicut, for some $\{s,t\} \in D$, and let $\widetilde{F}_i^0 \subseteq \delta^+_{\widetilde{G}}(\widetilde{W}_i)$ be arc sets such that $0 \leq r(a) \leq 2$ for all $a \in \widetilde{A}$. Let $\widetilde{F}_2$ (resp. $\widetilde{F}_1$) be the set of arcs such that $r(a) = 2$ (resp. $r(a) = 1$). Let $\widetilde{F}_1^2$ be the set of arcs $a \in \widetilde{F}_1$ for which there is another arc $a' \in \widetilde{F}_1$ which corresponds to the same edge of E, and let E_2 be the set of the corresponding edges. The inequality of type (5.28) associated with this configuration can be written as

$$\sum_{a \in \widetilde{F}_2} y(a) + \sum_{e \in E_2} x(e) + \sum_{a \in \widetilde{F}_1 \setminus \widetilde{F}_1^2} y(a) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}_i^0|}{2} \right\rceil. \tag{5.29}$$

As it will turn out, inequalities (5.29) may define facets under certain conditions and will be useful for solving the kHNDP using a Branch-and-Cut algorithm (Chapter 6).

5.6.1.2 Aggregated cut inequalities for kHNDP$_{Cu}(G,D)$

The aggregated cut inequalities can be defined for the polytope kHNDP$_{Cu}(G,D)$ in a similar way. Let $\widetilde{G}_{st} = (\widetilde{V}_{st}, \widetilde{A}_{st})$, $\{s,t\} \in D$, be the directed graphs associated with G and $\{s,t\} \in D$ in Formulation (5.10). Let $\{\{s_1,t_1\},...,\{s_q,t_q\}\}$ be a subset of demands. Consider a family of node sets $\{\widetilde{W}_1^{s_1t_1},...,\widetilde{W}_{p_1}^{s_1t_1},...,\widetilde{W}_1^{s_qt_q},...,\widetilde{W}_{p_q}^{s_qt_q}\}$, with $p_i \geq 1$, for all $i \in \{1,...,q\}$ and $p = \sum_{i=1}^{q} p_i \geq 2$, where $\widetilde{W}_j^{s_it_i}$, $j = 1,...,p_i$, induces an s_it_i-dicut in $\widetilde{G}_{st}$. Let $\widetilde{F}_j^{s_it_i,0} \subseteq \delta^+_{\widetilde{G}_{s_it_i}}(\widetilde{W}_j^{s_it_i})$. Let $\widetilde{F}^{s_it_i} = \bigcup_{i=1}^{p_i}[\delta^+_{\widetilde{G}_{s_it_i}}(\widetilde{W}_j^{s_it_i}) \setminus \widetilde{F}_j^{s_it_i,0}]$ for every $i \in \{1,...,q\}$, and for a given arc $a \in \widetilde{A}_{s_it_i}$, $i = 1,...,q$, we let $r_{s_it_i}(a)$ be the number of sets $\delta^+_{\widetilde{G}_{s_it_i}}(\widetilde{W}_j^{s_it_i}) \setminus \widetilde{F}_j^{s_it_i,0}$ containing arc a. If a does not belong to any of these sets, then $r_{s_it_i}(a) = 0$. Given an edge $e \in E$ and an arc subset $\widetilde{U}_i \subseteq \widetilde{A}_{s_it_i}$, we let $r'(e,\widetilde{U}_i) = \sum_{a \in \widetilde{A}_{s_it_i}(e) \cap \widetilde{U}_i} r_{s_it_i}(a)$.

The inequalities below are valid for $k\text{HNDP}_{Cu}(G, D)$

$$y_{s_it_i}(\delta^+_{\widetilde{G}_{s_it_i}}(\widetilde{W}^{s_it_i}_j)) \geq k \text{ for } j = 1, ..., p_i,\ i = 1, ..., q,$$
$$-y_{s_it_i}(a) \geq -1 \qquad \text{for } a \in \widetilde{F}^{s_it_i}_j,\ j = 1, ..., p_i,\ i = 1, ..., q,$$

By adding the inequalities, we get

$$\sum_{i=1}^{q}\left(\sum_{a\in\widetilde{F}^{s_it_i}} r_{s_it_i}(a)y_{s_it_i}(a)\right) \geq kp - \sum_{i=1}^{q}\sum_{j=1}^{p_i}|\widetilde{F}^{s_it_i,0}_j|.$$

Let $\widetilde{F}^{s_it_i,1}$ (resp. $\widetilde{F}^{s_it_i,2}$) be the set of arcs $a \in \widetilde{F}^{s_it_i}$ having $r_{s_it_i}(a)$ odd (resp. even). The inequality above can then be written as

$$\sum_{i=1}^{q}\left(\sum_{a\in\widetilde{F}^{s_it_i,1}} r_{s_it_i}(a)y_{s_it_i}(a) + \sum_{a\in\widetilde{F}^{s_it_i,2}} r_{s_it_i}(a)y_{s_it_i}(a)\right) \geq kp - \sum_{i=1}^{q}\sum_{j=1}^{p_i}|\widetilde{F}^{s_it_i,0}_j|. \quad (5.30)$$

Now we let $\widetilde{F}^{s_it_i,1}_2 \subseteq \widetilde{F}^{s_it_i,1}$, $i = 1, ..., q$, be the arc sets such that, for every edge $e \in E$ associated with an arc of $\widetilde{F}^{s_it_i,1}_2$, $\sum_{i=1}^{q} r'(e, \widetilde{F}^{s_it_i,1}_2)$ is even. If E_2 denotes the set of edges corresponding to the arcs of $\widetilde{F}^{s_it_i,1}_2$, $i = 1, ..., q$, then by adding inequality (5.30) and the inequalities

$$r_{s_it_i}(a)x(e) \geq r_{s_it_i}(a)y_{s_it_i}(a) \text{ for all } a \in \widetilde{F}^{s_it_i,1}_2 \text{ where } e \text{ corresponds to } a,$$

we get

$$\sum_{i=1}^{q}\left(\sum_{a\in\widetilde{F}^{s_it_i,1}\setminus\widetilde{F}^{s_it_i,1}_2} r_{s_it_i}(a)y_{s_it_i}(a) + \sum_{a\in\widetilde{F}^{s_it_i,2}} r_{s_it_i}(a)y_{s_it_i}(a)\right) +$$
$$\sum_{e\in E_2}\left(\sum_{i=1}^{q} r'(e, \widetilde{F}^{s_it_i,1}_2)\right)x(e) \geq kp - \sum_{i=1}^{q}\sum_{j=1}^{p_i}|\widetilde{F}^{s_it_i,0}_j|. \quad (5.31)$$

Finally, by dividing inequality (5.31) by 2 and rounding up the right hand side of the

resulting inequality, we obtain

$$\sum_{i=1}^{q}\left(\sum_{a\in\widetilde{F}^{s_it_i,1}\setminus\widetilde{F}_2^{s_it_i,1}}\frac{r_{s_it_i}(a)+1}{2}y_{s_it_i}(a)+\sum_{a\in\widetilde{F}^{s_it_i,2}}\frac{r_{s_it_i}(a)}{2}y_{s_it_i}(a)\right)+$$
$$\sum_{e\in E_2}\frac{\sum_{i=1}^{q}r'(e,\widetilde{F}_2^{s_it_i,1})}{2}x(e)\geq\left\lceil\frac{kp-\sum_{i=1}^{q}\sum_{j=1}^{p_i}|\widetilde{F}_j^{s_it_i}|}{2}\right\rceil. \tag{5.32}$$

We then have the following result.

Theorem 5.6.2 *Inequality (5.32) is valid for $kHNDP_{Cu}(G,D)$.*

Inequalities (5.32) will be also called *aggregated cut inequalities.*

We are also going to specify a special case for inequalities (5.32). These inequalities will be util in the Branch-and-Cut algorithm based on the Cut formulation (see Chapter 6). Let $\{\widetilde{W}_1^{s_1t_1},...,\widetilde{W}_{p_1}^{s_1t_1},...,\widetilde{W}_1^{s_qt_q},...,\widetilde{W}_{p_q}^{s_qt_q}\}$, with $p_i\geq 1$, for $i=1,...,q$, and $p=\sum_{i=1}^{q}p_i\geq 2$, be a family of node sets such that $\widetilde{W}_j^{s_it_i}$ induces s_it_i-dicut of $\widetilde{G}_{s_it_i}$, $i=1,...,q$. Let $\widetilde{F}_j^{s_it_i,0}\subseteq\delta^+_{\widetilde{G}_{s_it_i}}(\widetilde{W}_j^{s_it_i})$ be arc sets and $\widetilde{F}^{s_it_i}=\bigcup_{i=1}^{p}[\delta^+_{\widetilde{G}_{s_it_i}}(\widetilde{W}_j^{s_it_i})\setminus\widetilde{F}_j^{s_it_i,0}]$. Suppose that $0\leq r_{s_it_i}(a)\leq 2$ for all $a\in\widetilde{A}_{s_it_i}$, $i=1,...,q$. Let $\widetilde{F}^{s_it_i,2}$ be the set of arcs of $\widetilde{F}^{s_it_i}$ having $r_{s_it_i}(a)=2$ and $\widetilde{F}^{s_it_i,1}$ the set of arcs of $\widetilde{F}^{s_it_i}$ having $r_{s_it_i}(a)=1$. Let $\widetilde{F}_2^{s_it_i,1}$ be the subset of arcs $a\in\widetilde{F}^{s_it_i,1}$ such that there exists another arc $a'\in\widetilde{F}^{s_it_i,1}$ which corresponds to the same edge of E, and let E_2 be the set of the corresponding edges.

Then the inequality (5.32) induced by this configuration can be written as

$$\sum_{i=1}^{q}\left(\sum_{a\in\widetilde{F}^{s_it_i,2}}y_{s_it_i}(a)+\sum_{a\in\widetilde{F}^{s_it_i,1}\setminus\widetilde{F}_2^{s_it_i,1}}y_{s_it_i}(a)\right)+\sum_{e\in E_2}x(e)\geq\left\lceil\frac{kp-\sum_{i=1}^{q}\sum_{j=1}^{p_i}|\widetilde{F}_j^{s_it_i}|}{2}\right\rceil. \tag{5.33}$$

5.6.1.3 Lifting procedure for aggregated cut inequalities

In what follows we define a lifting procedure for the aggregated cut inequalities for both Aggregated and Cut formulations, (5.29) and (5.33). This will permit to extend these inequalities to a more general class of valid inequalities.

Consider first the polytope $k\text{HNDP}_{Ag}(G, D)$. The lifting procedure is given in the following theorem.

Theorem 5.6.3 *Let $G = (V, E)$ be an undirected graph, $D \subseteq V \times V$ and $\widetilde{G} = (\widetilde{V}, \widetilde{A})$ be the directed graph associated with G in the Aggregated formulation. Let*

$$\sum_{e \in E} \alpha(e)x(e) + \sum_{a \in \widetilde{A}} \beta(a)y(a) \geq \gamma$$

be an inequality of type (5.29) induced by a family of node sets $\Pi = \{\widetilde{W}_1, ..., \widetilde{W}_p\}$ and arc sets $\widetilde{F}_i^0 \subseteq \delta_i^0$, $p \geq 2$, which is valid for $kHNDP_{Ag}(G, D)$. Let $G' = (V, E \cup E')$ be a graph obtained by adding to G an edge set E and let $\widetilde{G}' = (\widetilde{V}, \widetilde{A} \cup \widetilde{A}')$ be the directed graph associated with G' in the Aggregated formulation ($\widetilde{A}'$ is the set of arcs corresponding to the edges of E'). Then, the inequality

$$\sum_{e \in E} \alpha(e)x(e) + \sum_{a \in \widetilde{A}} \beta(a)y(a) + \sum_{a \in \widetilde{A}'} \left\lceil \frac{q(a)}{2} \right\rceil y(a) \geq \gamma, \tag{5.34}$$

is valid for $kHNDP_{Ag}(G', D)$, where $q(a)$ is the number of dicuts $\delta^+_{\widetilde{G}'}(\widetilde{W}_i)$ containing the arc a, for all $a \in \widetilde{A}'$.

Proof. W.l.o.g., we will suppose that $E' = \{e_0\}$. The proof is similar in the case where more than one edge are added to G. Also, for more clarity, we will consider that only one arc, say a_0, is associated with e_0 in $\widetilde{G}'$, that we will consider that $\widetilde{A}' = \{a_0\}$.

We are going to show that for every solution $(\overline{x}, \overline{y}) \in k\text{HNDP}_{Ag}(G, D)$,

$$\sum_{e \in E} \alpha(e)\overline{x}(e) + \sum_{a \in \widetilde{A}} \beta(a)\overline{y}(a) + \left\lceil \frac{q(a_0)}{2} \right\rceil \overline{y}(a_0) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}_i^0|}{2} \right\rceil.$$

First, let $\Delta(x, y) = \alpha x + \beta y$, that is

$$\Delta(x, y) = \sum_{a \in \widetilde{F}_2} y(a) + \sum_{e \in E_2} x(e) + \sum_{a \in \widetilde{F}_1 \setminus \widetilde{F}_1^2} y(a),$$

where $\widetilde{F}_2$, $\widetilde{F}_1$, $\widetilde{F}_1^2$ and E_2 are the arc and edge sets involved in $\alpha x + \beta y \geq \gamma$. The lifted inequality can hence be written as

$$\Delta(x,y) + \left\lceil \frac{q(a_0)}{2} \right\rceil \overline{y}(a_0) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}_i^0|}{2} \right\rceil.$$

If $\overline{y}(a_0) = 0$, then obviously the restriction of $(\overline{x}, \overline{y})$ to E and $\widetilde{A}$ is in $k\text{HNDP}_{Ag}(G, D)$. Thus, $\Delta(\overline{x}, \overline{y}) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}_i^0|}{2} \right\rceil$, and hence

$$\Delta(\overline{x}, \overline{y}) + \left\lceil \frac{q(a_0)}{2} \right\rceil \overline{y}(a_0) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}_i^0|}{2} \right\rceil.$$

Now suppose that $\overline{y}(a_0) = 1$. We have that

$$\begin{aligned}
\sum_{i=1}^{p} \overline{y}(\delta_{\widetilde{G}}^+(\widetilde{W}_i) \setminus \widetilde{F}_i^0) &= \sum_{i=1}^{p} \overline{y}(\delta_{\widetilde{G}}^+(\widetilde{W}_i)) - \overline{y}(\widetilde{F}_i^0) \\
&= 2\sum_{a \in \widetilde{F}_2} \overline{y}(a) + \sum_{a \in \widetilde{F}_1^2} \overline{y}(a) + \sum_{a \in \widetilde{F}_1 \setminus \widetilde{F}_1^2} \overline{y}(a) \\
&\leq 2\sum_{a \in \widetilde{F}_2} \overline{y}(a) + 2\sum_{e \in E_2} \overline{x}(e) + \sum_{a \in \widetilde{F}_1 \setminus \widetilde{F}_1^2} \overline{y}(a) \\
&= 2\Delta(\overline{x}, \overline{y}) - \sum_{a \in \widetilde{F}_1 \setminus \widetilde{F}_1^2} \overline{y}(a)
\end{aligned}$$

Thus we get

$$\begin{aligned}
\Delta(\overline{x}, \overline{y}) &\geq \frac{1}{2}\left[\sum_{i=1}^{p} \overline{y}(\delta_{\widetilde{G}}^+(\widetilde{W}_i)) - \sum_{i=1}^{p} \overline{y}(\widetilde{F}_i^0) + \sum_{a \in \widetilde{F}_1 \setminus \widetilde{F}_1^2} \overline{y}(a)\right] \\
&\geq \frac{1}{2}\left[\sum_{i=1}^{p} \overline{y}(\delta_{\widetilde{G}}^+(\widetilde{W}_i)) - \sum_{i=1}^{p} \overline{y}(\widetilde{F}_i^0)\right]
\end{aligned}$$

$$\Delta(\overline{x},\overline{y}) \geq \left\lceil \frac{\sum_{i=1}^{p} \overline{y}(\delta^+_{\widetilde{G}}(\widetilde{W}_i)) - \sum_{i=1}^{p} |\widetilde{F}_i^0|}{2} \right\rceil. \tag{5.35}$$

If $\widetilde{W}_i$, $i = 1, ..., q(a_0)$, are the node sets of Π such that the dicut $\delta^+_{\widetilde{G}'}(\widetilde{W}_i)$ contains a_0, then we have that

$$\begin{array}{rcll} \overline{y}(\delta^+_{\widetilde{G}}(\widetilde{W}_i)) & = & \overline{y}(\delta^+_{\widetilde{G}'}(\widetilde{W}_i)) - \overline{y}(a_0), & i = 1, ..., q(a_0), \\ \overline{y}(\delta^+_{\widetilde{G}}(\widetilde{W}_i)) & = & \overline{y}(\delta^+_{\widetilde{G}'}(\widetilde{W}_i)), & i = q(a_0)+1, ..., p. \end{array}$$

As $(\overline{x},\overline{y})$ induces a solution of kHNDP$_{Ag}$ on G', we have that $\overline{y}(\delta^+_{\widetilde{G}'}(\widetilde{W}_i)) \geq k$, $i = 1, ..., p$. Moreover, since $\overline{y}(a_0) = 1$, we have that

$$\overline{y}(\delta^+_{\widetilde{G}}(\widetilde{W}_i)) \geq k-1, \ i = 1, ..., q(a_0). \tag{5.36}$$

Thus, from (5.35) and (5.36), we obtain

$$\Delta(\overline{x},\overline{y}) \geq \left\lceil \frac{k(p-q(a_0)) + (k-1)q(a_0) - \sum_{i=1}^{p} |\widetilde{F}_i^0|}{2} \right\rceil,$$

$$\Delta(\overline{x},\overline{y}) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}_i^0| - q(a_0)}{2} \right\rceil,$$

$$\Delta(\overline{x},\overline{y}) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}_i^0|}{2} \right\rceil - \left\lceil \frac{q(a_0)}{2} \right\rceil.$$

Therefore, since $\overline{y}(a_0) = 1$, we get

$$\Delta(\overline{x},\overline{y}) + \left\lceil \frac{q(a_0)}{2} \right\rceil \overline{y}(a_0) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}_i^0|}{2} \right\rceil,$$

which ends the proof of the theorem. □

Now we give a lifting procedure for aggregated cut inequalities (5.33) when the Cut formulation is considered. This procedure is similar to that introduced for inequalities (5.29) for the Aggregated formulation. It is given in the theorem below.

Theorem 5.6.4 *Let $G = (V, E)$ be an undirected graph, $D \subseteq V \times V$ and $\widetilde{G}_{st}$ be the directed graph associated with G and a demand $\{s, t\} \in D$ in the cut formulation, for all $\{s, t\} \in D$. Let*

$$\sum_{e \in E} \alpha(e)x(e) + \sum_{i=1}^{q} \sum_{a \in \widetilde{A}_{s_i t_i}} \beta_{s_i t_i}(a) y_{s_i t_i}(a) \geq \gamma,$$

be an inequality of type (5.33) induced by a demand set $\{\{s_1, t_1\}, ..., \{s_q, t_q\}\}$, a family of node sets $\{\widetilde{W}_1^{s_1 t_1}, ..., \widetilde{W}_{p_1}^{s_1 t_1}, ..., \widetilde{W}_1^{s_q t_q}, ..., \widetilde{W}_{p_q}^{s_q t_q}\}$, with $p_i \geq 1$, for all $i \in \{1, ..., q\}$ and $p = \sum_{i=1}^{q} p_i \geq 2$, and arc sets $\widetilde{F}_j^{s_i t_i, 0} \subseteq \delta_{\widetilde{G}_{s_i t_i}}(\widetilde{W}_j^{s_i t_i})$, $j = 1, ..., p_i$, $i = 1, ..., q$. Let $G' = (V, E \cup E')$ and $\widetilde{G}'_{st} = (\widetilde{V}_{st}, \widetilde{A}_{st} \cup \widetilde{A}'_{st})$ be the directed graph associated with G' in the Cut formulation, for all $\{s, t\} \in D$ ($\widetilde{A}'_{st}$ is the set of arcs corresponding to the edges of E').
The inequality

$$\sum_{e \in E} \alpha(e)x(e) + \sum_{i=1}^{q} \sum_{a \in \widetilde{A}_{s_i t_i}} \beta_{s_i t_i}(a) y_{s_i t_i}(a) + \sum_{i=1}^{q} \sum_{a \in \widetilde{A}'_{s_i t_i}} \left\lceil \frac{q_{s_i t_i}(a)}{2} \right\rceil y_{s_i t_i}(a) \geq \gamma \qquad (5.37)$$

is valid for kHNDP$_{Cu}(G', D)$, where $q_{s_i t_i}(a)$ is the number of dicuts $\delta^+_{\widetilde{G}'_{s_i t_i}}(\widetilde{W}_j^{s_i t_i})$ containing the arc a, for every $a \in \widetilde{A}'_{s_i t_i}$, $i = 1, ..., p$.

Proof. Similar to that of Theorem 5.6.3. □

The next classes of inequalities apply only on the variable $x \in \mathbb{R}^E$ and are valid for kHNDP$_{Ag}(G, D)$, kHNDP$_{Cu}(G, D)$, kHNDP$_{NA}(G, D)$ and kHNDP$_{PA}(G, D)$.

5.6.2 Double cut inequalities

In the following we introduce a class of inequalities that are valid for the kHNDP polytopes for $L \geq 2$ and $k \geq 2$. They are given by the following theorem.

Theorem 5.6.5 *Let $\{s,t\}$ be a demand, $i_0 \in \{0,...,L\}$ and $\Pi = \{V_0,...,V_{i_0-1},V^1_{i_0},V^2_{i_0},V_{i_0+1},...,V_{L+1}\}$ a family of node sets of V such that $\pi = (V_0,...,V_{i_0-1},V^1_{i_0},V^2_{i_0}\cup V_{i_0+1},V_{i_0+2},...,V_{L+1})$ induces a partition of V. Suppose that*

1. *$V^1_{i_0}\cup V^2_{i_0}$ induces an $s_{j_1}t_{j_1}$-cut of G with $\{s_{j_1},t_{j_1}\}\in D$ and $s_{j_1}\in V^1_{i_0}$ or $t_{j_1}\in V^1_{i_0}$ (note that s_{j_1} and t_{j_1} cannot be simultaneously in $V^1_{i_0}$ and are not in $V^2_{i_0}$. Also note that $V^2_{i_0}$ may be empty);*
2. *V_{i_0+1} induces an $s_{j_2}t_{j_2}$-cut of G with $\{s_{j_2},t_{j_2}\}\in D$ (note that j_1 and j_2 may be equal);*
3. *π induces an L-st-path-cut of G with $s\in V_0$ (resp. $t\in V_0$) and $t\in V_{L+1}$ (resp. $s\in V_{L+1}$).*

Let $\overline{E} = [V_{i_0-1},V^1_{i_0}]\cup[V_{i_0+2},V^2_{i_0}\cup V_{i_0+1}]\cup\left(\bigcup_{k,l\notin\{i_0,i_0+1\},|k-l|>1}[V_k,V_l]\right)$ *and $F\subseteq\overline{E}$ such that $|F|$ and k have different parities.*

Let also $\hat{E} = (\bigcup_{i=0}^{i_0-2}[V_i,V_{i+1}])\cup(\bigcup_{i=i_0+2}^{L}[V_i,V_{i+1}])\cup F$. *Then, the inequality*

$$x(\delta(\pi)\setminus\hat{E}) \geq \left\lceil\frac{3k-|F|}{2}\right\rceil, \tag{5.38}$$

is valid for $kHNDP_{Ag}(G,D)$, $kHNDP_{Cu}(G,D)$, $kHNDP_{NA}(G,D)$ and $kHNDP_{PA}(G,D)$ (recall that $\delta(\pi)$ is the set of edges of the E having their endnodes in different elements of π).

Proof. Let T be the L-st-path-cut of G induced by the partition π. As T is an L-st-path-cut, and $V^1_{i_0}\cup V^2_{i_0}$ and V_{i_0+1} induce st-cut with $\{s,t\}\in\{\{s_{j_1},t_{j_1}\},\{s_{j_2},t_{j_2}\}\}$, by Lemma 5.6.1, the inequalities below are valid for the kHNDP polytopes

$$\begin{aligned}
&x(T)\geq k,\\
&x(\delta(V^1_{i_0}\cup V^2_{i_0}))\geq k,\\
&x(\delta(V_{i_0+1}))\geq k,\\
&-x(e)\geq -1 \qquad \text{for all } e\in F,\\
&x(e)\geq 0 \qquad \text{for all } e\in\overline{E}\setminus F.
\end{aligned}$$

By summing these inequalities, dividing by 2 and rounding up the right hand side, we obtain inequality (5.38). □

Inequalities of type (5.38) are called *double cut inequalities*. They generalize those introduced by Huygens and Mahjoub [73] for the kHNDP when $k = 2$. We discuss in the following special cases for these inequalities. This concerns the case where $L \in \{2, 3\}$ and $i_0 = 0$.

The set of edges having a positive coefficient in inequality (5.38) plus the edges of F is called a *double cut*. Figure 5.4 gives an example for $L = 3$ and $i_0 = 0$.

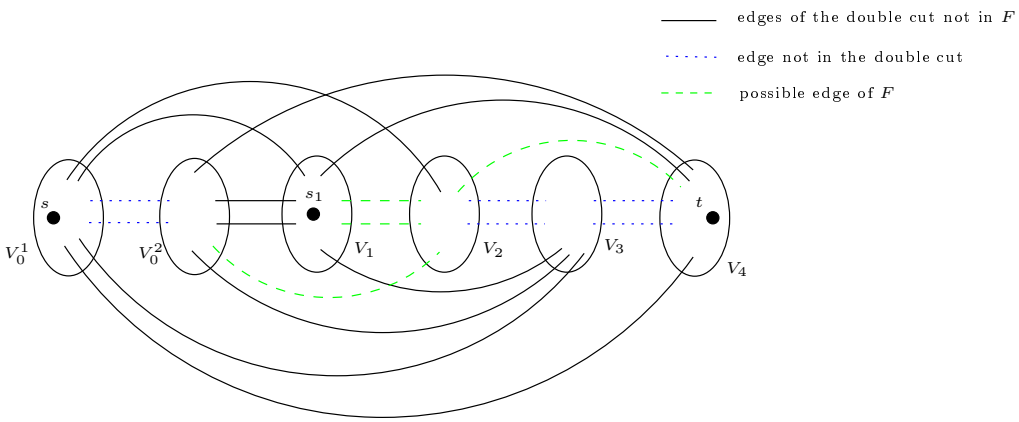


Figure 5.4: A double cut with $L = 3$ and $i_0 = 0$

Let $L = 2$, $\{s, t\} \in D$ and $\Pi = \{V_0^1, V_0^2, V_1, V_2, V_3\}$ be a family of node sets of V such that $\pi = (V_0^1, V_0^2 \cup V_1, V_2, V_3)$ induces a 2-st-path-cut, and V_1 induces a valid s_1t_1-cut in G, for some $\{s_1, t_1\} \in D$. If $F \subseteq [V_0^2 \cup V_1, V_2]$ is chosen such that $|F|$ and k have different parities, then the double cut inequality induced by Π and F in this case can be written as

$$x([V_0^1, V_1 \cup V_2 \cup V_3]) + x([V_0^2, V_1 \cup V_3]) + x([V_1, V_3]) + x([V_0^2 \cup V_1, V_2] \setminus F) \geq \left\lceil \frac{3k - |F|}{2} \right\rceil. \tag{5.39}$$

Now let $L = 3$, $\{s, t\} \in D$ and $\Pi = \{V_0^1, V_0^2, V_1, V_2, V_3, V_4\}$ be a family of node sets of V such that $\pi = (V_0^1, V_0^2 \cup V_1, V_2, V_3, V_4)$ induces a 3-st-path-cut, and V_1 induces a valid s_1t_1-cut in G. If $F \subseteq [V_0^2 \cup V_1 \cup V_4, V_2]$ is chosen such that $|F|$ and k have different

parities, then the double cut inequality induced by Π and F can be written as

$$x([V_0^1, V_1 \cup V_2 \cup V_3 \cup V_4]) + x([V_0^2, V_1 \cup V_3 \cup V_4]) + x([V_1, V_3 \cup V_4]) + x([V_0^2 \cup V_1 \cup V_4, V_2] \setminus F) \geq \left\lceil \frac{3k - |F|}{2} \right\rceil. \quad (5.40)$$

As it will turn out, inequalities (5.39) and (5.40) are very effective in the Branch-and-Cut algorithms we developed for the problem.

5.6.3 Triple path-cut inequalities

Here is a further class of valid inequalities. They also generalizes inequalities given by Huygens and Mahjoub [73]. We distinguish the cases where $L = 2$ and $L = 3$. We have the following theorem.

Theorem 5.6.6 *i) Let $L = 2$ and $\{V_0, V_1, V_2, V_3^1, V_3^2, V_4^1, V_4^2\}$ be a family of node sets of V such that $(V_0, V_1, V_2, V_3^1 \cup V_3^2, V_4^1 \cup V_4^2)$ induces a partition of V and there exist two demands $\{s_1, t_1\}$ and $\{s_2, t_2\}$ with $s_1, s_2 \in V_0$, $t_1 \in V_3^2$ and $t_2 \in V_4^2$. The sets V_3^1 and V_4^1 may be empty and s_1 and s_2 may be the same. Let also $V_3 = V_3^1 \cup V_3^2$, $V_4 = V_4^1 \cup V_4^2$ and $F \subseteq [V_3^2, V_1 \cup V_4^1] \cup [V_3^1, V_4^2]$ such that $|F|$ and k have different parities. Then, the inequality*

$$2x([V_0, V_2]) + x([V_0, V_3 \cup V_4]) + x([V_4^2, V_1 \cup V_3^2]) + x(([V_3^2, V_1 \cup V_4^1] \cup [V_3^1, V_4^2]) \setminus F) \geq \left\lceil \frac{3k - |F|}{2} \right\rceil \quad (5.41)$$

is valid for $kHNDP_{Ag}(G, D)$, $kHNDP_{Cu}(G, D)$, $kHNDP_{NA}(G, D)$ and $kHNDP_{PA}(G, D)$.

ii) Let $L = 3$ and $(V_0, ..., V_3, V_4^1, V_4^2, V_5^1, V_5^2)$ be a family of node sets of V such that $(V_0, ..., V_3, V_4^1 \cup V_4^2, V_5^1 \cup V_5^2)$ induces a partition of V and there exist two demands $\{s_1, t_1\}$ and $\{s_2, t_2\}$ with $s_1, s_2 \in V_0$, $t_1 \in V_4^2$ and $t_2 \in V_5^2$. The sets V_4^1 and V_5^1 may be empty and s_1 and s_2 may be the same. Let also $V_4 = V_4^1 \cup V_4^2$, $V_5 = V_5^1 \cup V_5^2$ and $F \subseteq [V_2, V_4^2] \cup [V_3, V_4 \cup V_5]$ such that $|F|$ and k have different parities. Then, the inequality

$$2x([V_0, V_2]) + 2x([V_0, V_3]) + 2x([V_1, V_3]) + x([V_0 \cup V_1, V_4 \cup V_5]) + x([V_4, V_5]) + x([V_2, V_5^2]) + x(([V_2, V_4^2] \cup [V_3, V_4 \cup V_5]) \setminus F) \geq \left\lceil \frac{3k - |F|}{2} \right\rceil \quad (5.42)$$

is valid for $kHNDP_{Ag}(G, D)$, $kHNDP_{Cu}(G, D)$, $kHNDP_{NA}(G, D)$ and $kHNDP_{PA}(G, D)$.

Proof.
i) Let T_1 be the 2-s_1t_1-path-cut induced by the partition $(V_0, V_1 \cup V_4, V_2 \cup V_3^1, V_3^2)$ and T_2 and T_3 the 2-s_2t_2-path-cuts induced by the partitions $(V_0, V_1 \cup V_3, V_2 \cup V_4^1, V_4^2)$ and $(V_0, V_1, V_2 \cup V_3 \cup V_4^1, V_4^2)$, respectively. By Lemma 5.6.1, the following inequalities are valid for the kHNDP polytopes

$$\begin{aligned}
&x(T_1) \geq k,\\
&x(T_2) \geq k,\\
&x(T_3) \geq k,\\
&-x(e) \geq -1, \text{ for all } e \in F,\\
&x(e) \geq 0, \text{ for all } e \in ([V_3^2, V_1 \cup V_4^1] \cup [V_3^1, V_4^2]) \setminus F.
\end{aligned}$$

By adding these inequalities, dividing by 2 and rounding up the right hand side, we get inequality (5.41).

ii) Let T_1 be the 3-s_1t_1-path-cut induced by the partition $(V_0, V_1 \cup V_5, V_2, V_3 \cup V_4^1, V_4^2)$, and T_2 and T_3 be the 3-s_2t_2-path-cuts induced by the partitions $(V_0, V_1 \cup V_4, V_2, V_3 \cup V_5^1, V_5^2)$ and $(V_0, V_1, V_2, V_3 \cup V_4 \cup V_5^1, V_5^2)$, respectively. By Lemma 5.6.1, the following inequalities are valid for the kHNDP polytopes

$$\begin{aligned}
&x(T_1) \geq k,\\
&x(T_2) \geq k,\\
&x(T_3) \geq k,\\
&-x(e) \geq -1, && \text{for all } e \in F,\\
&x(e) \geq 0, && \text{for all } e \in ([V_2, V_4^2] \cup [V_3, V_4 \cup V_5]) \setminus F.
\end{aligned}$$

By adding these inequalities, dividing by 2 and rounding up the right hand side, we get inequality (5.42). □

Inequalities of type (5.41) and (5.42) will be called *triple path-cut inequalities*. The set of edges having a positive coefficient in inequality (5.41) ((5.42)) plus the edges of F will be called a *triple path-cut* (see Figure 5.5 for an example with $L = 2$).

In the next two sections, we describe two more classes of inequalities.

5.6.4 Steiner-partition inequalities

Let $(V_0, V_1, ..., V_p)$, $p \geq 2$, be a partition of V such that $V_0 \subseteq V \setminus R_D$, where R_D is the set of terminal nodes of G, and for all $i \in \{1, ..., p\}$ there is a demand $\{s, t\} \in D$ such

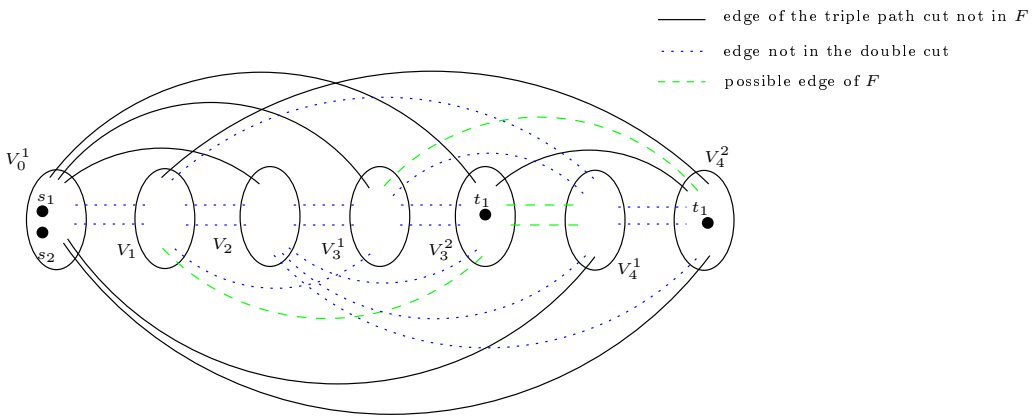


Figure 5.5: A triple path-cut with $L = 2$

that V_i induces an st-cut of G. Note that V_0 may be empty. Such a partition is called a *Steiner-partition*. With a Steiner-partition, we associate the inequality

$$x(\delta(V_0, V_1, ..., V_p)) \geq \left\lceil \frac{kp}{2} \right\rceil. \tag{5.43}$$

Inequalities of type (5.43) will be called *Steiner-partition inequalities*. We have the following result.

Theorem 5.6.7 *Inequality (5.43) is valid for $kHNDP_{Ag}(G, D)$, $kHNDP_{Cu}(G, D)$, $kHNDP_{NA}(G, D)$ and $kHNDP_{PA}(G, D)$.*

Proof. By Lemma 5.6.1, the inequalities below are valid for the kHNDP polytopes

$$x(\delta(V_i)) \geq k, \text{ for } i = 1, ..., p,$$
$$x(e) \geq 0, \text{ for all } e \in \delta(V_0).$$

By adding them, we obtain

$$2x(\delta(V_0, ..., V_p)) \geq kp.$$

By dividing by 2 and rounding up the right hand side, we get inequality (5.43). □

Inequality (5.43) expresses the fact that, in a solution of the kHNDP, the multicut induced by a Steiner-partition $(V_0, V_1, ..., V_p)$, $p \geq 2$, must contain at least $\lceil \frac{kp}{2} \rceil$ edges, since there must exist k edge-disjoint paths between every pair of nodes $\{s, t\} \in D$.

5.6.5 Steiner-SP-partition inequalities

Let $\pi = (V_1, ..., V_p)$, $p \geq 3$, be a partition of V such that the graph $G_\pi = (V_\pi, E_\pi)$ is series-parallel (G_π is the subgraph of G induced by π). Suppose that $V_\pi = \{v_1, ..., v_p\}$ where v_i is the node of G_π corresponding to the set V_i, $i = 1, ..., p$. The partition π is said to be a *Steiner-SP-partition* if and only if π is a Steiner-partition and either

1. $p = 3$ or
2. $p \geq 4$ and there exists a node $v_{i_0} \in V_\pi$ incident to exactly two nodes v_{i_0-1} and v_{i_0+1} such that the partitions π_1 and π_2 obtained from π by contracting respectively the sets V_{i_0}, V_{i_0-1} and V_{i_0}, V_{i_0+1} are themselves Steiner-SP-partitions.

The procedure to check if a partition is a Steiner-SP-partition is recursive. It stops when the partition obtained after the different contractions is either a Steiner-partition and of size three or it is not a Steiner-partition.

In the following theorem, we give necessary and sufficient condition for a Steiner-partition to be a Steiner-SP-partition. Remind that the demand graph is denoted by $G_D = (R_D, E_D)$, where R_D is the set of terminal nodes of G. The edge set E_D is obtained by adding an edge between two nodes of R_D if and only if $\{u, v\} \in D$.

Theorem 5.6.8 *Let $\pi = (V_1, ..., V_p)$, $p \geq 3$, be a partition of V such that G_π is series-parallel. The partition π is a Steiner-SP-partition of G if and only if the subgraph of G_D induced by π is connected.*

Proof. First observe that, as π is a SP-partition of G, one can obtain from π a two-size partition by applying repeatidly the following operation. Let $\pi^j = (V_1^j, ..., V_{p_j}^j)$ be a SP-partition of G. Suppose that $V_{i_0}^j$, for some i_0, is incident to exactly two elements $V_{i_0-1}^j$ and $V_{i_0+1}^j$. Then, the operation consists in contracting the sets $V_{i_0-1}^j$ and $V_{i_0}^j$ and consider the partition $\pi^{j+1} = (V_1^{j+1}, ..., V_{p_{j+1}}^{j+1})$ where

$$
\begin{aligned}
&V_i^{j+1} = V_i^j \text{ for } i = 1, ..., i_0 - 2,\\
&V_{i_0-1}^{j+1} = V_{i_0-1}^j \cup V_{i_0}^j,\\
&V_i^{j+1} = V_{i+1}^j \text{ for } i = i_0, ..., p_j - 1.
\end{aligned}
$$

Note that the new partition π^{j+1} induces a SP-partition of G and that we have $p-2$ iterations to obtained a two-size partition from π.

Now, we have that π is not a Steiner-SP-partition if and only if there exists an integer $q \leq p-2$ such that the partition $\pi^q = (V_1^q, ..., V_{p_q}^q)$, obtained by application of the above operation, is not a Steiner-partition, that is the node set $V_{i_0}^q$ of π^q obtained by the contraction procedure to the partition π^{q-1} is such that $\delta_{G_D}(V_{i_0}^q) = \emptyset$. Thus, if $V_{i_1}, ..., V_{i_r}$, $r \geq 2$, are the node sets of π that have been reduced to $V_{i_0}^q$ during the different steps of the contraction procedure, then we have that $\delta_{G_D}(\bigcup_{i=1}^{r} V_{i_r}) = \emptyset$. Therefore, the subgraph of G_d induced by π is not connected, which ends the proof. □

As a consequence of Theorem 5.6.8, if the demand graph is connected (this is the case when, for instance, all the demands are rooted in the same node), then every Steiner-partition of V inducing a series-parallel subgraph of G is a Steiner-SP-partition of V.

With a Steiner-SP-partition $(V_1, ..., V_p)$, $p \geq 3$, we associate the following inequality

$$x(\delta(V_1, ..., V_p)) \geq \left\lceil \frac{k}{2} \right\rceil p - 1. \tag{5.44}$$

Inequalities of type (5.44) will be called *Steiner-SP-partition inequalities*. We have the following.

Theorem 5.6.9 *Inequality (5.44) is valid for $kHNDP_{Ag}(G, D)$, $kHNDP_{Cu}(G, D)$, $kHNDP_{NA}(G, D)$ and $kHNDP_{PA}(G, D)$.*

Proof. Let $\pi = (V_1, ..., V_p)$, $p \geq 3$ be a Steiner-SP-partition. The proof is by induction on p. If $p = 3$, then, as π is a Steiner-partition, the inequality

$$x(\delta(V_1, V_2, V_3)) \geq \left\lceil \frac{3k}{2} \right\rceil = 3 \left\lceil \frac{k}{2} \right\rceil - 1$$

is valid.

Now suppose that every inequality (5.44) induced by a Steiner-SP-partition of p elements, $p \geq 3$, is valid for the kHNDP polytopes and consider a Steiner-SP-partition $\pi = (V_1, ..., V_p, V_{p+1})$. As G_π is series-parallel, there exists a node set V_{i_0} of π which is incident to exactly two elements of π, say V_{i_0-1} and V_{i_0+1}. We let $F_1 = [V_{i_0}, V_{i_0-1}]$ and $F_2 = [V_{i_0}, V_{i_0+1}]$. Since π is a Steiner-SP-partition and hence is a Steiner-partition, by

Lemma 5.6.1, V_{i_0} induces a valid st-cut inequality, for some $\{s,t\} \in D$. Hence we have that

$$x(F_1) + x(F_2) \geq k.$$

W.l.o.g., we will suppose that

$$x(F_1) \geq \left\lceil \frac{k}{2} \right\rceil. \tag{5.45}$$

Consider the partition $\pi' = (V_1, ..., V_{i_0-2}, V_{i_0-1} \cup V_{i_0}, V_{i_0+1}, ..., V_{p+1})$. As π is a Steiner-SP-partition containing more than three elements, π' is also a Steiner-SP-partition which contains p elements. Thus, by the induction hypothesis, the Steiner-SP-partition inequality induced by π', that is

$$x(\delta(V_1, ..., V_{i_0-2}, V_{i_0-1} \cup V_{i_0}, V_{i_0+1}, ..., V_{p+1})) \geq \left\lceil \frac{k}{2} \right\rceil p - 1 \tag{5.46}$$

is valid. By summing the inequalities (5.45) and (5.46), we get

$$x(\delta(V_1, ..., V_p, V_{p+1})) \geq \left\lceil \frac{k}{2} \right\rceil (p+1) - 1,$$

which ends the proof of the theorem. □

Inequality (5.44) expresses the fact that in a solution of the kHNDP the multicut induced by a Steiner-SP-partition contains at least $\lceil \frac{k}{2} \rceil p - 1$ edges, since this solution contains k edge-disjoint paths between every pair of nodes $\{s,t\} \in D$.

Chopra [21] described a lifting procedure for inequalities (2.27) for the kECSP. This procedure can be easily extended, for the kHNDP, to inequalities of type (5.44). It is described as follows. Let $G = (V, E)$ be a graph and $k \geq 3$ an odd integer. Let $G' = (V, E \cup E')$ be a graph obtained from G by adding an edge set E'. Let $\pi = (V_1, ..., V_p)$ be a Steiner-SP-partition of G. Then the following inequality is valid for kHNDP$_{Ag}(G, D)$, kHNDP$_{Cu}(G, D)$, kHNDP$_{NA}(G, D)$ and kHNDP$_{PA}(G, D)$

$$x(\delta_G(V_1, ..., V_p)) + \sum_{e \in E' \cap \delta_{G'}(V_1, ..., V_p)} a(e)x(e) \geq \left\lceil \frac{k}{2} \right\rceil p - 1, \tag{5.47}$$

where $a(e)$ is the length (in terms of edges) of a shortest path in G_π between the endnodes of e, for all $e \in E' \cap \delta_{G'}(V_1, ..., V_p)$.

We will call inequalities of type (5.47) *lifted Steiner-SP-partition inequalities*.

In the next section, we investigate conditions under which aggregated cut, double cut and triple path-cut inequalities define facets of the kHNDP polytopes.

5.7 Facets

Throughout this section, we consider a complete graph $G = (V, E)$ and suppose that $|V| \geq k + 2$.

The first result concerns necessary conditions for the aggregated cut inequalities (5.29) to define facets for $k\text{HNDP}_{Ag}(G, D)$. To this end, we first give the following lemma.

Lemma 5.7.1 *Consider an inequality of type (5.29) induced by a family of node sets* $\Pi = \{\widetilde{W}_1, ..., \widetilde{W}_p\}$, $p \geq 2$, *and arc subsets* $\widetilde{F}_i^0 \subseteq \delta_{\widetilde{G}}^+(\widetilde{W}_i)$, $i = 1, ..., p$. *Let* $\widetilde{F}_2$, $\widetilde{F}_1$, $\widetilde{F}_1^2$ *and* E_2 *be the arc and edge sets involved in this inequality. Then (5.29) can be written as*

$$\sum_{i=1}^{p} y(\delta^+(\widetilde{W}_i)) + 2\sum_{e\in E_2} x(e) - \sum_{a\in \widetilde{F}_1^2} y(a) + \sum_{i=1}^{p}(|\widetilde{F}_i^0| - y(\widetilde{F}_i^0)) + \sum_{a\in \widetilde{F}_1\setminus\widetilde{F}_1^2} y(a) \geq kp + 1. \tag{5.48}$$

Moreover, (5.29) is tight for a solution $(x_0, y_0) \in kHNDP_{Ag}(G, D)$ *if and only if one of the following conditions holds*

i)

$$2\sum_{e\in E_2} x_0(e) - \sum_{a\in \widetilde{F}_1^2} y_0(a) + \sum_{i=1}^{p}(|\widetilde{F}_i^0| - y_0(\widetilde{F}_i^0)) + \sum_{a\in \widetilde{F}_1\setminus\widetilde{F}_1^2} y_0(a) = 1 \tag{5.49}$$

and $y_0(\delta^+(\widetilde{W}_i)) = k$, *for* $i = 1, ..., p$;

ii)

$$2\sum_{e\in E_2} x_0(e) - \sum_{a\in \widetilde{F}_1^2} y_0(a) + \sum_{i=1}^{p}(|\widetilde{F}_i^0| - y_0(\widetilde{F}_i^0)) + \sum_{a\in \widetilde{F}_1\setminus\widetilde{F}_1^2} y_0(a) = 0 \tag{5.50}$$

and there exists $i_0 \in \{1, ..., p\}$ *such that* $y_0(\delta^+(\widetilde{W}_i)) = k$, *for* $i \in \{1, ..., p\} \setminus \{i_0\}$ *and* $y_0(\delta^+(\widetilde{W}_{i_0})) = k + 1$.

Proof. First we show that $\alpha x + \beta y \geq \gamma$ is equivalent to (5.48). As kp and $\sum_{i=1}^{p} |\widetilde{F}_i^0|$ have different parities, $\alpha x + \beta y \geq \gamma$ is equivalent to

$$2\sum_{e\in E_2} x(e) + 2\sum_{a\in \widetilde{F}_2} y(a) + 2\sum_{a\in \widetilde{F}_1\setminus \widetilde{F}_1^2} y(a) \geq kp - \sum_{i=1}^{p} |\widetilde{F}_i^0| + 1. \tag{5.51}$$

From the st-dicuts induced by the sets $\widetilde{W}_i$, we have that

$$\begin{aligned}\sum_{i=1}^{p} y(\delta^+(\widetilde{W}_i)\setminus \widetilde{F}_i^0) &= 2\sum_{a\in \widetilde{F}_2} y(a) + \sum_{a\in \widetilde{F}_1^2} y(a) + \sum_{a\in \widetilde{F}_1\setminus \widetilde{F}_1^2} y(a),\\ &= 2\sum_{a\in \widetilde{F}_2} y(a) + 2\sum_{e\in E_2} x(e) - 2\sum_{e\in E_2} x(e) + \sum_{a\in \widetilde{F}_1^2} y(a) + \sum_{a\in \widetilde{F}_1\setminus \widetilde{F}_1^2} y(a).\end{aligned}$$

Toghether with (5.51), we get

$$\sum_{i=1}^{p} y(\delta^+(\widetilde{W}_i)\setminus \widetilde{F}_i^0) + 2\sum_{e\in E_2} x(e) - \sum_{a\in \widetilde{F}_1^2} y(a) + \sum_{a\in \widetilde{F}_1\setminus \widetilde{F}_1^2} y(a) \geq kp - \sum_{i=1}^{p} |\widetilde{F}_i^0| + 1. \tag{5.52}$$

By combining (5.52) and $y(\delta^+(\widetilde{W}_i)\setminus \widetilde{F}_i^0) = y(\delta^+(\widetilde{W}_i)) - y(\widetilde{F}_i^0)$, $i = 1, ..., p$, we get (5.48).

Now consider a solution $(x_0, y_0) \in k\text{HNDP}_{Ag}(G, D)$ satisfying (5.29) with equality. By the previous result, we have that

$$\sum_{i=1}^{p} y_0(\delta^+(\widetilde{W}_i)) + \sum_{i=1}^{p}(|\widetilde{F}_i^0| - y_0(\widetilde{F}_i^0)) + 2\sum_{e\in E_2} x_0(e) - \sum_{a\in \widetilde{F}_1^2} y_0(a) + \sum_{a\in \widetilde{F}_1\setminus \widetilde{F}_1^2} y_0(a) = kp + 1. \tag{5.53}$$

As (x_0, y_0) induces a solution of the kHNDP, we have that $y_0(\delta^+(\widetilde{W}_i)) \geq k$, $i = 1, ..., p$. Therefore, $\sum_{i=1}^{p} y_0(\delta^+(\widetilde{W}_i)) \geq kp$, and hence,

$$\sum_{i=1}^{p}(|\widetilde{F}_i^0| - y_0(\widetilde{F}_i^0)) + 2\sum_{e\in E_2} x_0(e) - \sum_{a\in \widetilde{F}_1^2} y_0(a) + \sum_{a\in \widetilde{F}_1\setminus \widetilde{F}_1^2} y_0(a) \leq 1. \tag{5.54}$$

If (5.54) is satisfied with equality, then, clearly $y_0(\delta^+(\widetilde{W}_i)) = k$, $i = 1, ..., p$. If not, then, as $y_0(\delta^+(\widetilde{W}_i)) \geq k$, $i = 1, ..., p$, this yields $y_0(\delta^+(\widetilde{W}_{i_0})) = k + 1$ for some $i_0 \in \{1, ..., p\}$ and $y_0(\delta^+(\widetilde{W}_i)) = k$, for $i \in \{1, ..., p\} \setminus \{i_0\}$.

Conversely, if (5.54) is tight for (x_0, y_0) and $y_0(\delta^+(\widetilde{W}_i)) = k$ for all $i \in \{1, ..., p\}$, then clearly, (5.48) is tight for (x_0, y_0) and hence $\alpha x + \beta y \geq \gamma$ is tight for (x_0, y_0). If (5.54) is not tight for (x_0, y_0), that is

$$\sum_{i=1}^{p}(|\widetilde{F}_i^0| - y_0(\widetilde{F}_i^0)) + 2\sum_{e\in E_2} x_0(e) - \sum_{a\in \widetilde{F}_1^2} y_0(a) + \sum_{a\in \widetilde{F}_1\setminus\widetilde{F}_1^2} y_0(a) = 0,$$

and $y_0(\delta^+(\widetilde{W}_{i_0})) = k + 1$ for some $i_0 \in \{1, ..., p\}$ and $y_0(\delta^+(\widetilde{W}_{i_0})) = k$ for $i \in \{1, ..., p\} \setminus \{i_0\}$, then clearly, (5.48) is also tight for (x_0, y_0). Thus, $\alpha x + \beta y \geq \gamma$ is tight for (x_0, y_0). □

Corollary 5.7.1 *Consider an inequality of type (5.29) induced by a family of node sets* $\{\widetilde{W}_1, ..., \widetilde{W}_p\}$, $p \geq 2$, *and arc subsets* $\widetilde{F}_i^0 \subseteq \delta^+_{\widetilde{G}}(\widetilde{W}_i)$, $i = 1, ..., p$. *Let* $\widetilde{F}_2$, $\widetilde{F}_1$, $\widetilde{F}_1^2$ *and* E_2 *be the arc and edge sets involved in this inequality. If (5.29) is tight for a solution* (x_0, y_0) *of* $kHNDP_{Ag}(G, D)$ *then,*

$$2\sum_{e\in E_2} x_0(e) - \sum_{a\in \widetilde{F}_1^2} y_0(a) + \sum_{i=1}^{p}(|\widetilde{F}_i^0| - y_0(\widetilde{F}_i^0)) + \sum_{a\in \widetilde{F}_1\setminus\widetilde{F}_1^2} y_0(a) \leq 1. \tag{5.55}$$

Theorem 5.7.1 *Let* $\Pi = \{\widetilde{W}_1, ..., \widetilde{W}_p\}$, $p \geq 2$, *be a family of node sets of* $\widetilde{V}$ *such that each set* $\widetilde{W}_i$, $i = 1, ..., p$, *induces an* s_it_i*-dicut of* $\widetilde{G}$, *for some* $\{s_i, t_i\} \in D$, *and* $\widetilde{F}_i^0 \subseteq \delta^+_{\widetilde{G}}(\widetilde{W}_i)$. *Suppose that every arc of* $\widetilde{A}$ *belongs to at most two sets* $\delta^+_{\widetilde{G}}(\widetilde{W}_i) \setminus \widetilde{F}_i^0$. *Then, the aggregated cut inequality (5.29) induced by* Π *and* $\widetilde{F}_i^0$, $i = 1, ..., p$, *defines a facet of* $kHNDP_{Ag}(G, D)$ *different from the trivial and* s_it_i*-dicut inequalities, only if for all* $i \in \{1, ..., p\}$, *one of the following conditions holds*

1. $|\widetilde{W}_i \cap S_D| = |(\widetilde{V} \setminus \widetilde{W}_i) \cap T_D| = 1$;
2. $|\widetilde{W}_i \cap S_D| \geq 2$ *and for all* $s \in (\widetilde{W}_i \setminus \{s_i\}) \cap S_D$, $[s, \widetilde{V} \setminus \widetilde{W}_i] = \emptyset$;
3. $|(\widetilde{V} \setminus \widetilde{W}_i) \cap T_D| \geq 2$ *and for all* $t \in [(\widetilde{V} \setminus \widetilde{W}_i) \setminus \{t_i\}] \cap T_D$, $[\widetilde{W}_i, t] = \emptyset$.

Proof. Let us denote by $\alpha x + \beta y \geq \gamma$ the inequality (5.29) induced by Π and $\widetilde{F}_i^0$, $i = 1, ..., p$, and suppose that it defines a facet of $k\text{HNDP}_{Ag}(G, D)$. We will show that $|\widetilde{W}_i \cap S_D| = 1$, for $i = 1, ..., p$. The proof follows the same lines for $|(\widetilde{V} \setminus \widetilde{W}_i) \cap T_D| = 1$. Also the proof for 2) and 3) is similar.

Suppose on the contrary that there exists $i_0 \in \{1, ..., p\}$ such that $\widetilde{W}_{i_0}$ induces an st-dicut of $\widetilde{G}$ and that $(\widetilde{W}_{i_0} \setminus \{s\}) \cap S_D \neq \emptyset$. Let s' be a node of $(\widetilde{W}_{i_0} \setminus \{s\}) \cap S_D$ and suppose that $[s', \widetilde{V} \setminus \widetilde{W}_{i_0}] \neq \emptyset$ (see Figure 5.6).

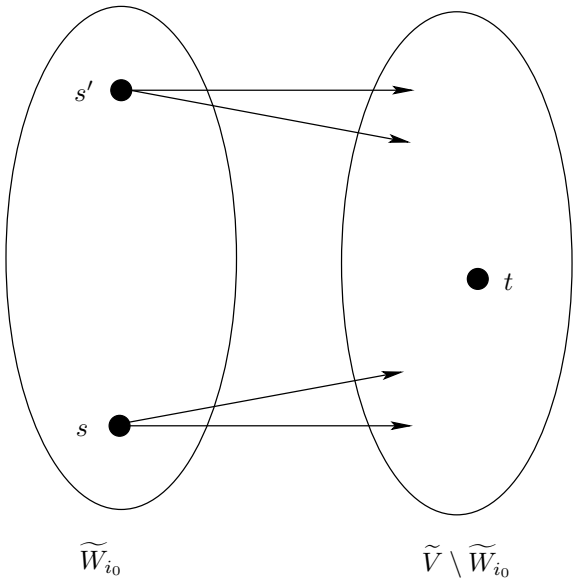


Figure 5.6: A set $\widetilde{W}_{i_0}$ containing two nodes of S

First observe that $\delta_{\widetilde{G}}(\widetilde{W}'_{i_0}) = \delta_{\widetilde{G}}(\widetilde{W}'_{i_0}) \setminus [s', \widetilde{V} \setminus \widetilde{W}_{i_0}]$ and that two arcs of $[s', \widetilde{V} \setminus \widetilde{W}_{i_0}]$ do not correspond to the same edge of E.

Let $\widetilde{H}_0 = \widetilde{F}_2 \cap [s', \widetilde{V} \setminus \widetilde{W}_{i_0}]$ and $\widetilde{H}_1 = (\widetilde{F}_1 \setminus \widetilde{F}_1^2) \cap [s', \widetilde{V} \setminus \widetilde{W}_{i_0}]$. Also let $\widetilde{H}_2 = \widetilde{F}_1^2 \cap [s', \widetilde{V} \setminus \widetilde{W}_{i_0}]$, $\widetilde{H}_3$ be the set of arcs of $\widetilde{F}_1^2$ corresponding to the same edges as the arcs of $\widetilde{H}_2$. Let E_0 be edge set corresponding to the arcs of $\widetilde{H}_2$ and $\widetilde{H}_3$. Consider now the aggregated cut inequality induced by $\{\widetilde{W}'_1, ..., \widetilde{W}'_p\}$ and $\widetilde{F}_i^{0'}$, $i = 1, ..., p$, where $\widetilde{W}'_i = \widetilde{W}_i$, $\widetilde{F}_i^{0'} = \widetilde{F}_i^0$, for $i \in \{1, ..., p\} \setminus \{i_0\}$, and $\widetilde{W}'_{i_0} = \widetilde{W}_{i_0} \setminus \{s'\}$, $\widetilde{F}_i^{0'} = \widetilde{F}_i^0 \setminus [s', \widetilde{V} \setminus \widetilde{W}_{i_0}]$. Let $\widetilde{F}'_2$, $\widetilde{F}'_1$, $\widetilde{F}_1^{2'}$ and E'_2 be the set of arcs and edges involved in this inequality. By the above observation, as the arcs of $\widetilde{H}_3$ correspond to those of $\widetilde{H}_2$, we have that $\widetilde{H}_3 \cap [s', \widetilde{V} \setminus \widetilde{W}_{i_0}] = \emptyset$. Also, by the same observation, no arc of $\widetilde{H}_0$ may correspond to

an arc of $\widetilde{H}_2$ and $\widetilde{H}_3$. Thus, we have that

$$\begin{aligned}
&\widetilde{F}_2' = \widetilde{F}_2 \setminus \widetilde{H}_0,\\
&\widetilde{F}_1^{2'} = \widetilde{F}_1^2 \setminus (\widetilde{H}_2 \cup \widetilde{H}_3),\\
&\widetilde{F}_1' \setminus \widetilde{F}_1^{2'} = [(\widetilde{F}_1 \setminus \widetilde{F}_1^2) \setminus \widetilde{H}_1] \cup \widetilde{H}_0 \cup \widetilde{H}_3.\\
&E_2' = E_2 \setminus E_0.
\end{aligned}$$

Therefore, the inequality (5.29) induced by $\{\widetilde{W}_1', ..., \widetilde{W}_p'\}$ and $\widetilde{F}_i^{0'}$, $i = 1, ..., p$, can be written as

$$\sum_{a \in \widetilde{F}_2 \setminus \widetilde{H}_0} y(a) + \sum_{e \in E_2 \setminus E_0} x(e) + \sum_{a \in (\widetilde{F}_1 \setminus \widetilde{F}_1^2) \setminus \widetilde{H}_1} + \sum_{a \in \widetilde{H}_0} y(a) + \sum_{a \in \widetilde{H}_3} y(a) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}_i^{0'}|}{2} \right\rceil. \tag{5.56}$$

By summing up inequality (5.56) and the inequalities

$$x(e) \geq y(a), \text{ for all } a \in \widetilde{H}_3, \text{ where } e \text{ is the edge of } E_0 \text{ corresponding to } a. \tag{5.57}$$

$$y(a) \geq 0, \text{ for all } a \in \widetilde{H}_1, \tag{5.58}$$

we get

$$\sum_{a \in \widetilde{F}_2} y(a) + \sum_{e \in E_2} x(e) + \sum_{a \in \widetilde{F}_1 \setminus \widetilde{F}_1^2} y(a) \geq \left\lceil \frac{kp - \sum_{i=1}^{p} |\widetilde{F}_i^{0'}|}{2} \right\rceil. \tag{5.59}$$

Clearly if $\widetilde{F}_{i_0} \cap [s', \widetilde{V} \setminus \widetilde{W}_{i_0}] = \emptyset$, then $\widetilde{F}_{i_0}' = \widetilde{F}_{i_0}$ and inequality (5.59) is the same as $\alpha x + \beta y \geq \gamma$. Thus $\alpha x + \beta y \geq \gamma$ is redundant with respect to (5.56)-(5.58), and hence cannot define a facet of kHNDP$_{Ag}(G, D)$. If $\widetilde{F}_{i_0} \cap [s', \widetilde{V} \setminus \widetilde{W}_{i_0}] \neq \emptyset$, then the right hand side of inequality (5.59) is greater than that of $\alpha x + \beta y \geq \gamma$. Thus, $\alpha x + \beta y \geq \gamma$ is dominated by (5.56)-(5.58), and hence cannot define a facet of kHNDP$_{Ag}(G, D)$. □

The next theorems give necessary conditions for the double cut and triple path-cut inequalities to define facets of the kHNDP polytopes. Before each theorem, we will give a technical lemma which will be useful to prove the theorem.

Lemma 5.7.2 *Let $\alpha x \geq \gamma$ be a double cut inequality induced by a family of node sets $\Pi = (V_0^1, V_0^2, V_1, ..., V_{L+1})$ of V, $F \subseteq E$ and $\{s,t\} \in D$ with $s \in V_0^1$ and $t \in V_{L+1}$ (here $i_0 = 0$). Then, $\alpha x \geq \gamma$ can be written as*

$$x(T) + x(\delta(V_0^1 \cup V_0^2)) + x(\delta(V_1)) + x(\overline{E} \setminus F) + |F| - x(F) \geq 3k + 1, \tag{5.60}$$

where T is the L-st-path-cut induced by the partition $(V_0^1, V_0^2 \cup V_1, V_2, ..., V_{L+1})$.

Moreover, $\alpha x \geq \gamma$ is tight for a solution x_0 of kHNDP$_{Ag}$, kHNDP$_{Cut}$, kHNDP$_{NA}$, kHNDP$_{PA}$, where $x_0 \in \mathbb{R}^E$, if and only if one of the following conditions holds.

i) $x_0(\overline{E} \setminus F) + |F| - x_0(F) = 1$ *and* $x_0(T) = x_0(\delta(V_0^1 \cup V_0^2)) = x_0(V_1) = k$;

ii) $x_0(\overline{E} \setminus F) + |F| - x_0(F) = 0$ *and*

- *a)* $x_0(T) = k+1$, $x_0(\delta(V_0^1 \cup V_0^2)) = k$ *and* $x_0(V_1) = k$;
- *b)* $x_0(T) = k$, $x_0(\delta(V_0^1 \cup V_0^2)) = k+1$ *and* $x_0(V_1) = k$;
- *c)* $x_0(T) = k$, $x_0(\delta(V_0^1 \cup V_0^2)) = k$ *and* $x_0(V_1) = k+1$;

Proof. W.l.o.g., we will consider the polytope kHNDP$_{Ag}(G,D)$. The proof is similar for The proof is similar for kHNDP$_{Cut}(G,D)$, kHNDP$_{NA}(G,D)$ and kHNDP$_{PA}(G,D)$.

Let H denote the double cut induced by Π. The inequality $\alpha x \geq \gamma$ is equivalent to

$$x(H \setminus \overline{E}) + x(\overline{E} \setminus F) \geq \frac{3k - |F| + 1}{2}.$$

This implies that

$$2x(H \setminus \overline{E}) + 2x(\overline{E}) - 2x(F) \geq 3k - |F| + 1. \tag{5.61}$$

From the L-st-path-cut T and cuts $\delta(V_0^1 \cup V_0^2)$ and $\delta(V_1)$, we have that

$$x(T) + x(\delta(V_0^1 \cup V_0^2)) + x(\delta(V_1)) = 2x(H \setminus \overline{E}) + x(\overline{E}). \tag{5.62}$$

By combining (5.61) and the (5.62), we get

$$x(T) + x(\delta(V_0^1 \cup V_0^2)) + x(\delta(V_1)) + x(\overline{E}) - 2x(F) \geq 3k - |F| + 1,$$

and hence

$$x(T) + x(\delta(V_0^1 \cup V_0^2)) + x(\delta(V_1)) + x(\overline{E} \setminus F) + |F| - x(F) \geq 3k + 1.$$

Therefore, $\alpha x \geq \gamma$ is equivalent to (5.60).

Now suppose that $\alpha x \geq \gamma$ is tight for (x_0, y_0). From the development above, we have that inequality (5.60) is also tight for (x_0, y_0), that is

$$x_0(T) + x_0(\delta(V_0^1 \cup V_0^2)) + x_0(\delta(V_1)) + x_0(\overline{E} \setminus F) + |F| - x_0(F) = 3k + 1.$$

Since by Lemma 5.6.1, $x_0(T) \geq k$, $x_0(\delta(V_0^1 \cup V_0^2)) \geq k$ and $x_0(\delta(V_1)) \geq k$, it is clear that $x_0(\overline{E} \setminus F) + |F| - x_0(F) \leq 1$. Hence, if $x_0(\overline{E} \setminus F) + |F| - x_0(F) = 1$, we have that $x_0(T) = x_0(\delta(V_0^1 \cup V_0^2)) = x_0(\delta(V_1)) = k$. If $x_0(\overline{E} \setminus F) + |F| - x_0(F) = 0$, then, clearly, either $x_0(T)$, $x_0(\delta(V_0^1 \cup V_0^2))$ or $x_0(\delta(V_1))$ is equal to $k+1$ and the others are equal to k.

Consider now a solution $(x_0, y_0) \in k\text{HNDP}_{Ag}(G, D)$ such that $x_0(\overline{E} \setminus F) + |F| - x_0(F) = 1$ and $x_0(T) = x_0(\delta(V_0^1 \cup V_0^2)) = x_0(\delta(V_1)) = k$. Then, clearly, inequality (5.60) is satisfied with equality, and hence, $\alpha x \geq \gamma$ is tight for (x_0, y_0). Similarly, if $x_0(\overline{E} \setminus F) + |F| - x_0(F) = 0$ and either $x_0(T)$, $x_0(\delta(V_0^1 \cup V_0^2))$ or $x_0(\delta(V_1))$ is equal to $k+1$ with the others equal to k, then (5.60) is satisfied with equality by x_0 and hence, $\alpha x \geq \gamma$ is tight for (x_0, y_0), which ends the proof of the lemma. □

Theorem 5.7.2 *Suppose that $L \geq 2$ and $k \geq 2$, and let $\{s, t\} \in D$.*
Let $\Pi = \{V_0^1, V_0^2, V_1, ..., V_{L+1}\}$ be a family of node sets of V and $F \subseteq E$ which induce a double cut of G with respect to $\{s, t\}$, $s \in V_0^1$ and $t \in V_{L+1}$ (here $i_0 = 0$). Then, the double cut inequality induced by Π and F defines a facet of kHNDP$_{Ag}(G, D)$, kHNDP$_{Cu}(G, D)$, kHNDP$_{NA}(G, D)$, kHNDP$_{PA}(G, D)$ different from the trivial inequalities and inequalities (5.1)-(5.2) only if the following conditions hold

i) $|V_0^1| = |V_{L+1}| = 1$;

ii) if $L = 3$, *then* $|[V_0^1, V_0^2 \cup V_1] \cup [V_3, V_4] \cup [V_0^1, V_4]| \geq k$.

Proof. The proof will be done for $k\text{HNDP}_{Ag}(G, D)$ as it is similar for $k\text{HNDP}_{Cu}(G, D)$, $k\text{HNDP}_{NA}(G, D)$ and $k\text{HNDP}_{PA}(G, D)$. We will denote by $\alpha x \geq \gamma$ the double cut inequality induced by Π and F. Let $\mathcal{F} = \{(x, y) \in k\text{HNDP}_{Ag}(G, D)$ such that $\alpha x = \gamma\}$ and let T denote the L-st-path-cut induced by the partition $(V_0^1, V_0^2 \cup V_1, V_2, ..., V_{L+1})$.

i) Let us denote by H the double cut induced by Π and F. Suppose first that $|V_0^1| \geq 2$. By considering the family of node sets $\Pi' = \{\{s\}, V_0^2 \cup V_0^1 \setminus \{s\}, V_1, ..., V_{L+1}\}$, the double

cut H' induced by Π' and F is such that $H = H' \cup [V_0^1 \setminus \{s\}, V_1]$. Thus, the double cut inequality induced by H is redundant with respect to

$$x(H' \setminus F) \geq \left\lceil \frac{3k - |F|}{2} \right\rceil$$
$$x(e) \geq, \text{ for all } e \in [V_0^1 \setminus \{s\}, V_1], \tag{5.63}$$

and hence, cannot define a facet.

ii) We will show that $\mathcal{F} \neq \emptyset$ only if ii) holds. As $\mathcal{F}$ defines a facet different from $x(\delta(V_0^1 \cup V_0^2)) \geq k$, there exists a solution $(\overline{x}, \overline{y}) \in \mathcal{F}$ such that $\overline{x}(\delta(V_0^1 \cup V_0^2)) \geq k+1$. Thus, by Lemma 5.7.2, $\overline{x}(T) = k$. Therefore, the graph induced by $\overline{x}$ contains exactly k edge-disjoint L-st-paths. Moreover, each L-st-path intersects T only once. Thus, by Lemma 4.2.2, we have that $|[V_0^1, V_{L+1}]| + |[V_0^1, V_0^2 \cup V_1]| + |[V_L, V_{L+1}]| \geq k$. □

Lemma 5.7.3 *Let $\alpha x \geq \gamma$ be a triple path-cut inequality induced by a family of node set $\Pi = \{V_0, ..., V_L, V_{L+1}^1, V_{L+1}^2, V_{L+2}^1, V_{L+2}^2\}$ and $F \subseteq E$. Then $\alpha x \geq \gamma$ can be written as*

$$x(T_1) + x(T_2) + x(T_3) + x(\overline{E} \setminus F) + |F| - x(F) \geq 3k + 1 \tag{5.64}$$

where T_1, T_2 and T_3 are the triple path-cuts induced by the partitions $(V_0, V_1 \cup V_4, V_2 \cup V_3^1, V_3^2)$, $(V_0, V_1 \cup V_3, V_2 \cup V_4^1, V_4^2)$ and $(V_0, V_1, V_2 \cup V_3 \cup V_4^1, V_4^2)$, respectively, and $\overline{E} = [V_3^2, V_1 \cup V_4^1] \cup [V_3^1, V_4^2]$ (resp. $\overline{E} = [V_2, V_4^2] \cup [V_3, V_4 \cup V_5]$) if $L = 2$ (resp. $L = 3$).

Moreover, $\alpha x \geq \gamma$ is tight for a solution x_0 of the kHNDP, where $x_0 \in \mathbb{R}^E$, if and only if one of the following inequalities holds

i) $x_0(\overline{E} \setminus F) + |F| - x_0(F) = 1$ and $x_0(T_1) = x_0(T_2) = x_0(T_3) = k$;

ii) $x_0(\overline{E} \setminus F) + |F| - x_0(F) = 0$ and, for some $i_0 \in \{1, 2, 3\}$, $x_0(T_{i_0}) = k + 1$ and $x_0(T_i) = k$ for $i \in \{1, 2, 3\} \setminus \{i_0\}$.

Proof. Similar to that of Lemma 5.7.2. □

Theorem 5.7.3 *Let $L \in \{2, 3\}$ and consider $\Pi = \{V_0, ..., V_L, V_{L+1}^1, V_{L+1}^2, V_{L+2}^1, V_{L+2}^2\}$ be a family of node sets of V and $F \subseteq E$ which induce a triple path-cut of G with respect to demands $\{s_1, t_1\}$ and $\{s_2, t_2\}$. Then, the triple path-cut inequality induced by Π and F defines a facet of $kHNDP_{Ag}(G, D)$, $kHNDP_{Cu}(G, D)$, $kHNDP_{NA}(G, D)$, $kHNDP_{PA}(G, D)$ only if the following conditions hold*

i) $V_0 \setminus \{s_1, s_2\} = \emptyset$;

ii) $|V^2_{L+1}| = 1$;

iii) $|V^2_{L+2}| = 1$;

iv) if $L = 3$, *then*

a) $|[\{s_1, s_2\}, V_1 \cup V^1_5 \cup \{t_2\}]| + |[V_3 \cup V^1_4, t_1]| + |[\{s_1, s_2\}, t_1]| \geq k$;

b) $|[\{s_1, s_2\}, V_1 \cup V^1_4 \cup \{t_1\}]| + |[V_3 \cup V^1_5, t_2]| + |[\{s_1, s_2\}, t_2]| \geq k$;

c) $|[\{s_1, s_2\}, V_1]| + |[V_3 \cup V^1_4 \cup \{t_1\} \cup V^1_5, t_2]| + |[\{s_1, s_2\}, t_2]| \geq k$.

Proof. For the proof of Conditions i)-iii), we will consider, w.l.o.g., that $L = 3$. We will denote by $\alpha x \geq \gamma$ the triple-cut inequality induced by Π and F.

i) Suppose that $V_0 \setminus \{s_1, s_2\} \neq \emptyset$ and denote by H the triple path-cut induced by Π and F. Consider the family of node sets $\Pi' = \{\{s_1, s_2\}, V_0 \setminus \{s_1, s_2\} \cup V_1, V_2, V_3, V^1_4, V^2_4, V^1_5, V^2_5\}$ and $F' = F$. If H' denotes the triple path-cut induced by Π' and F', we have that $H' = H \setminus [V_0 \setminus \{s_1, s_2\}, V_2]$. Thus, as $V_0 \setminus \{s_1, s_2\} \neq \emptyset$, inequality (5.42) induced by Π and F is redundant with respect to the inequalities

$$\begin{aligned} &2x([\{s_1, s_2\}, V_2]) + 2x([\{s_1, s_2\}, V_3]) + 2x([V_1 \cup (V_0 \setminus \{s_1, s_2\}), V_3]) + \\ &\qquad x([\{s_1, s_2\} \cup V_1 \cup (V_0 \setminus \{s_1, s_2\}), V_4 \cup V_5]) + x([V_4, V_5]) + x([V_2, V^2_5]) + \\ &\qquad\qquad x(([V_2, V^2_4] \cup [V_3, V_4 \cup V_5]) \setminus F) \geq \left\lceil \frac{3k - |F|}{2} \right\rceil, \\ &x(e) \geq 0, \text{ for all } e \in [V_0 \setminus \{s_1, s_2\}, V_2]. \end{aligned}$$

Therefore, the triple path-cut inequality induced by Π and F cannot define a facet of the kHNDP polytopes.

ii) Now we show that $|V^2_4| = 1$. Suppose on the contrary that $|V^2_4| \geq 2$ and let $\alpha x \geq \gamma$ denote the triple path-cut inequality induced by Π and F. Let $\Pi' = \{V_0, ..., V_3, V^1_4 \cup V^2_4 \setminus \{t_1\}, \{t_1\}, V^1_5, V^2_5\}$. First suppose that $F \cap [V_2, V^2_4 \setminus \{t_1\}] = \emptyset$ and let H' be the triple path-cut induced by Π' and F. As $F \cap [V_2, V^2_4 \setminus \{t_1\}] = \emptyset$, we have that $H' = H \setminus [V_2, V^2_4 \setminus \{t_1\}]$. If $\alpha' x \geq \gamma'$ denotes the triple path-cut inequality induced by Π' and F, then it is not hard to see that $\alpha'(e) = \alpha(e)$, for all $e \in H' \setminus F$, and that $\gamma' = \gamma$. Thus, $\alpha x \geq \gamma$ is redundant with respect to the following inequalities

$$\begin{aligned} &\alpha' x \geq \gamma, \\ &x(e) \geq 0, \text{ for all } e \in [V_2, V^2_4 \setminus \{t_1\}], \end{aligned}$$

and hence, cannot define a facet of the kHNDP poytopes.

If $F \cap [V_2, V_4^2 \setminus \{t_1\}] \neq \emptyset$, then we consider $F' = F \setminus (F \cap [V_2, V_4^2 \setminus \{t_1\}])$ and let $\alpha' x \geq \gamma'$ be the triple path-cut inequality induced by Π' and F'. Also let H' denotes this triple path-cut. As before, we have that $H' = H \setminus [V_2, V_4^2 \setminus \{t_1\}]$ and, for all $e \in H' \setminus F'$, $\alpha'(e) = \alpha(e)$. Moreover, $\gamma = \left\lceil \frac{3k-|F|}{2} \right\rceil$ and $\gamma' = \left\lceil \frac{3k-|F|+|F\cap[V_2,V_4^2\setminus\{t_1\}]|}{2} \right\rceil$. As $|F \cap [V_2, V_4^2 \setminus \{t_1\}]| \geq 1$, we have that $\gamma' \geq \gamma$. This implies that $\alpha x \geq \gamma$ is dominated by the inequalities

$$\begin{aligned} &\alpha' x \geq \gamma', \\ &x(e) \geq 0, \text{ for all } e \in [V_2, V_4^2 \setminus \{t_1\}] \setminus F. \end{aligned}$$

Thus, it cannot define a facet of the kHNDP poytopes.

iii) Suppose that $|V_5^2| \geq 2$. Consider $\Pi' = \{V_0, ..., V_3, V_4^1, V_4^2, V_5^1 \cup V_5^2 \setminus \{t_2\}, \{t_2\}\}$ and let H and H' denote the triple path-cuts induced by Π and F, and by Π' and F respectively. If $F \cap [V_3, V_5^2 \setminus \{t_2\}] = \emptyset$, then, clearly, $H' = H \setminus [V_2, V_5^2 \setminus \{t_2\}]$. If $F \cap [V_3, V_5^2 \setminus \{t_2\}] \neq \emptyset$, then it is also not hard to see that, as before, $H' = H \setminus [V_2, V_5^2 \setminus \{t_2\}]$.

This implies that the triple path-cut inequality induced by H is redundant with respect to that induced by H' and the inequalities $x(e) \geq 0$, for all $e \in [V_2, V_5^2 \setminus \{t_2\}]$. Thus, it cannot define a facet.

iv) To show that conditions iv) are necessary for $\alpha x \geq \gamma$ to define a facet, we show that the sets $\mathcal{F}_i = \{x \in \mathbb{R}^E$ such that x induces a solution of the kHNDP and $x(T_i) = k\}$, $i = 1, 2, 3$, are non empty only if conditions iv) are satisfied. As $\mathcal{F}$ is different from the inequality $x(e) \leq 0$ for some $e \in F$, there exists a solution $(\overline{x}, \overline{y}) \in \mathcal{F}$ such that $\overline{x}(e) = 0$. Thus, $|F| - \overline{x}(F) \geq 1$. By Lemma 5.7.3, this implies that $\overline{x}(\overline{E} \setminus F) + |F| - \overline{x}(F) = 1$ and hence, $\overline{x}(T_i) = k$, for $i = 1, 2, 3$. Therefore, from Lemma 4.2.2, we obtain that

$$\begin{aligned} &|[\{s_1, s_2\}, V_1 \cup V_5^1 \cup \{t_2\}]| + |[V_3 \cup V_4^1, t_1]| + |[\{s_1, s_2\}, t_1]| \geq k, \\ &|[\{s_1, s_2\}, V_1 \cup V_4^1 \cup \{t_1\}]| + |[V_3 \cup V_5^1, t_2]| + |[\{s_1, s_2\}, t_2]| \geq k, \\ &|[\{s_1, s_2\}, V_1]| + |[V_3 \cup V_4^1 \cup \{t_1\} \cup V_5^1, t_2]| + |[\{s_1, s_2\}, t_2]| \geq k, \end{aligned}$$

which ends the proof of the theorem. □

In the following chapter, we use all the results presented in this chapter to devise Branch-and-Cut and Branch-and-Cut-and-Price algorithms for the kHNDP. As it will turn out, these results will be particularly useful to develop effecient separation algorithms for the various inequalities we have presented here.

Chapter 6

Branch-and-Cut and Branch-and-Cut-and-Price Algorithms for the kHNDP

In this chapter we present Branch-and-Cut and Branch-and-Cut-and-Price algorithms we have devised to solve the kHNDP. In Sections 6.1 and 6.2, we will describe the framework of these algorithms. In Section 6.4, we will present some computational results and in Section 6.5 we give some concluding remarks.

In order to solve the kHNDP using Aggregated, Cut and Node-Arc formulations, we use a Branch-and-Cut algorithm. These formulations use a polynomial number of variables. For the Path-Arc formulation, we use a Branch-and-Cut-and-Price algorithm since this formulation uses an exponential number of variables. These algorithms are described in Sections 6.1 and 6.2. Section 6.3 describes the various separation routines used in both Branch-and-Cut and Branch-and-Cut-and-Price algorithms.

Here we recall some notations that will be used all along this chapter. Given an undirected graph $G = (V, E)$ and a demand set $D \subseteq V \times V$, the set of terminal nodes involved in a demand as source (resp. destination) node is denoted by S_D (resp. T_D). The set of terminal nodes is denoted by R_D. The demand graph $G_D = (R_D, E_D)$ is the undirected graph whose nodes are those of R_D and, for every demand $\{u, v\} \in D$, we add an edge uv in G_D. The directed graph associated with G in the Aggregated formulation is denoted by $\widetilde{G} = (\widetilde{V}, \widetilde{A})$ and the directed graphs associated with G in the separated formulations (Cut, Node-Arc and Path-Arc formulations) are denoted by $\widetilde{G}_{st} = (\widetilde{V}_{st}, \widetilde{A}_{st})$, $\{s, t\} \in D$.

Given a solution $x \in [0,1]^E$, the *support graph* $G(x) = (V, E(x))$ is the subgraph of G obtained by removing from G all the edges $e \in E$ such that $x(e) = 0$, that is $E(x) = \{e \in E \mid x(e) > 0\}$. Also, we let

$$E_0(x) = \{e \in E \mid x(e) = 0\},$$
$$E_1(x) = \{e \in E \mid x(e) = 1\},$$
$$E_f(x) = \{e \in E \mid 0 < x(e) < 1\}.$$

In a similar way, given a solution $y \in [0,1]^{\widetilde{A}}$, the support graph $\widetilde{G}(y) = (\widetilde{V}, \widetilde{A}(\overline{y}))$ is the subgraph of $\widetilde{G}$ obtained by removing from $\widetilde{G}$ all the arcs $a \in \widetilde{A}$ such that $y(a) = 0$, that is $\widetilde{A}(y) = \{a \in \widetilde{A} \mid y(a) = 0\}$. Also, we let

$$\widetilde{A}_0(x) = \{a \in \widetilde{A} \mid y(e) = 0\},$$
$$\widetilde{A}_1(x) = \{a \in \widetilde{A} \mid y(e) = 1\},$$
$$\widetilde{A}_f(x) = \{a \in \widetilde{A} \mid 0 < y(e) < 1\}.$$

Finally, for a demand $\{s,t\} \in$ and a solution $y_{st} \in [0,1]^{\widetilde{A}_{st}}$, the support graph is the graph $\widetilde{G}_{st}(y_{st}) = (\widetilde{V}_{st}, \widetilde{A}_{st}(y_{st}))$, is the graph such that $\widetilde{A}_{st}(\overline{y}_{st}) = \{a \in \widetilde{A}_{st} \mid y_{st}(a) > 0\}$. We let

$$\widetilde{A}^0_{st}(y_{st}) = \{a \in \widetilde{A}_{st} \mid y_{st}(a) = 0\},$$
$$\widetilde{A}^1_{st}(y_{st}) = \{a \in \widetilde{A}_{st} \mid y_{st}(a) = 1\},$$
$$\widetilde{A}^f_{st}(y_{st}) = \{a \in \widetilde{A}_{st} \mid 0 < y_{st}(a) < 1\}.$$

6.1 Branch-and-Cut algorithms for Aggregated, Cut and Node-Arc formulations

We first describe a Branch-and-Cut algorithm for the Aggregated formulation. To start the optimization, we consider the linear program given by the st-dicut inequalities induced by the node sets $\{s\}$, $\{s\} \cup N'$ and $\{s\} \cup N' \cup N''$, for all $s \in S_D$, toghether

with the linking and trivial inequalities. That is to say, we consider the program

$$\begin{aligned}
&\text{Min} \sum_{e\in E} c(e)x(e)\\
&\left.\begin{array}{l} y(\delta^+_{\widetilde{G}}(s)) \geq k,\\ y(\delta^+_{\widetilde{G}}(\{s\}\cup V_1)) \geq k,\\ y(\delta^+_{\widetilde{G}}(\{s\}\cup V_1\cup V_2)) \geq k, \end{array}\right\} \text{ for all } s\in S_D,\\
&y(a) \leq x(e), \text{ for all } a\in \widetilde{A}(e),\ e\in E,\\
&y(a) \geq 0, \text{ for all } a\in \widetilde{A},\\
&x(e) \leq 1, \text{ for all } e\in E.
\end{aligned}$$

The optimal solution $(\overline{x},\overline{y})$ of this LP is feasible for $k\text{HNDP}_{Ag}$ if and only if $(\overline{x},\overline{y})$ is integral and satisfies every st-dicut inequality, for all $\{s,t\}\in D$. If $(\overline{x},\overline{y})$ is not feasible for the problem, then we generate further valid inequalities for $k\text{HNDP}_{Ag}(G,D)$ that are violated by $(\overline{x},\overline{y})$. To do this, the algorithm tries to add in the current LP the following inequalities, in this order,

1. st-dicut inequalities,
2. aggregated cut inequalities,
3. double cut inequalities,
4. triple path-cut inequalities,
5. Steiner-partition inequalities,
6. Steiner-SP-partition inequalities.

For the Cut formulation, the optimization starts by considering the following linear program

$$\begin{aligned}
&\text{Min} \sum_{e\in E} c(e)x(e)\\
&\left.\begin{array}{l} y_{st}(\delta^+_{\widetilde{G}_{st}}(s)) \geq k,\\ y_{st}(\delta^+_{\widetilde{G}_{st}}(\{s\}\cup N_{st})) \geq k,\\ y_{st}(\delta^+_{\widetilde{G}_{st}}(\{s\}\cup N_{st}\cup N'_{st})) \geq k,\\ y_{st}(a) \leq x(e), \text{ for all } a\in \widetilde{A}_{st}(e),\ e\in E,\\ y_{st}(a) \geq 0, \text{ for all } a\in \widetilde{A}_{st}, \end{array}\right\} \text{ for all } \{s,t\}\in D,\\
&x(e) \leq 1, \text{ for all } e\in E.
\end{aligned}$$

Here also, the optimal solution $(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$ is feasible for kHNDP$_{Cu}$ if $(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$ is integral and satisfies every st-dicut inequality, for all $\{s,t\} \in D$. If $(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$ is not feasible for the problem, then we generate, as before, further valid inequalities for kHNDP$_{Cu}(G,D)$ that are violated by $(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$. For this, we look for the following inequalities, in this order,

1. st-dicut inequalities,
2. aggregated cut inequalities,
3. double cut inequalities,
4. triple path-cut inequalities,
5. Steiner-partition inequalities,
6. Steiner-SP-partition inequalities.

Now we describe the Branch-and-Cut algorithm for the Node-Arc formulation. The optimization starts by solving the linear relaxation of Formulation (5.15). As this formulation contains a polynomial number of variables and constraints, its linear relaxation can be solved using only one linear program,

$$\text{Min} \sum_{e \in E} c(e)x(e)$$
$$\text{sujected to}$$
$$(5.11) - (5.14).$$

The optimal solution $(\overline{x}, \overline{f}^{s_1t_1}, ..., \overline{f}^{s_dt_d})$ of this LP is feasible for kHNDP$_{NA}$ if it is integral. If this is not the case, we then try to add further inequalities that are valid for kHNDP$_{NA}(G,D)$ and violated by this solution. The inequalities that are considered here are the following, generated in this order,

1. double cut inequalities,
2. triple path-cut inequalities,
3. Steiner-partition inequalities,
4. Steiner-SP-partition inequalities.

6.2 A Branch-and-Cut-and-Price algorithm for Path-Arc formulation

The Branch-and-Cut-and-Price algorithm for the kHNDP starts by solving the linear relaxation of Formulation (5.20). As this formulation uses an exponential number of variables but a polynomial number of constraints, we use a column generation algorithm to solve its linear relaxation.

6.2.1 Column generation algorithm

Remind that the column generation algorithm starts by solving a linear program obtained from the linear relaxation of the Path-Arc formulation by considering a subset of variables which induce a feasible basis for the initial problem. For our purpose, we consider first the sets of st-dipaths $\mathcal{B}_{st} \subseteq \mathcal{P}_{st}$, $\{s,t\} \in D$, such that $|\mathcal{B}_{st}| \geq k$ and the paths of $\mathcal{B}_{st}$ are arc-disjoint. Note that the subgraph of $\widetilde{G}_{st}$ induced by the paths of $\mathcal{B}_{st}$ contains k arc-disjoint st-dipaths. By Corollary 5.2.1, the edge set corresponding to the arcs involved in the paths of $\mathcal{B}_{st}$, $\{s,t\} \in D$, induces a solution of the kHNDP, and, toghether with the sets $\mathcal{B}_{st}$, $\{s,t\} \in D$, induces a feasible solution for the linear relaxation of Formulation (5.20). Hence, we consider as initial set of variables those induced by the edge set E and the sets $\mathcal{B}_{st}$, $\{s,t\} \in D$. The first the linear program solved in the column generation algorithm is, therefore, the one obtained from the linear relaxation of Formulation (5.20) and these variables. This linear program is

$$\text{Min} \sum_{e \in E} c(e)x(e)$$

$$\sum_{\widetilde{P} \in \mathcal{B}_{st}} \mu^{st}(\widetilde{P}) \geq k, \tag{6.1}$$

$$\sum_{\widetilde{P} \in \mathcal{B}_{st}} \gamma^{st}_{\widetilde{P},a} \mu^{st}(\widetilde{P}) \leq x(e), \text{ for all } a \in \widetilde{A}_{st}(e), e \in E, \tag{6.2}$$

$$\mu^{st}(\widetilde{P}) \geq 0, \text{ for every } \widetilde{P} \in \mathcal{B}_{st}, \text{ and every } \{s,t\} \in D, \tag{6.3}$$

$$x(e) \leq 1, \text{ for all edge } e \in E. \tag{6.4}$$

At each iteration, the algorithm tries to generate new columns, that is to add to $\mathcal{B}_{st}$, $\{s,t\} \in D$, directed paths $\widetilde{P} \in \mathcal{P}_{st} \setminus \mathcal{B}_{st}$ such that the variable $\mu^{st}(\widetilde{P})$ has a negative reduced cost. This is done by solving the so-called *satellite problem* which consists in finding, for all $\{s,t\} \in D$, a path $\widetilde{P}^*$ such that $c_r(\widetilde{P}^*) = \min\{c_r(\widetilde{P}) \mid \widetilde{P} \in \mathcal{P}_{st}\}$ and $c_r(\widetilde{P}^*) < 0$, where $c_r(\widetilde{P})$ is the reduced cost of the variable $\mu^{st}(\widetilde{P})$.

The reduced cost $c_r(\widetilde{P})$ is computed using the dual optimal solution. Let λ_0^{st} and λ_a^{st}, $a \in \widetilde{A}_{st}$, be the dual variables associated with inequalities (6.1) and (6.2), respectively. Then, given a path $\widetilde{P} \in \mathcal{P}_{st}$, for some $\{s,t\} \in D$, the reduced cost of the variable $\mu^{st}(\widetilde{P})$ is given by

$$c_r(\widetilde{P}) = \lambda_0^{st} + \sum_{a \in \widetilde{A}_{st}} \gamma_{\widetilde{P},a}^{st} \lambda_a^{st} = \lambda_0^{st} + \sum_{a \in \widetilde{P}} \lambda_a^{st}.$$

Thus, the satellite problem reduces to find a shortest st-dipath in the graph $\widetilde{G}_{st}$, for all $\{s,t\} \in D$, with respect to lengths λ_a^{st} on arc $a \in \widetilde{A}_{st}$. If a shortest st-dipath of $\widetilde{G}_{st}$, say $\widetilde{P}^*$, is such that $\sum_{a \in \widetilde{P}^*} \lambda_a^{st} < -\lambda_0^{st}$, then $c_r(\widetilde{P}^*) < 0$. If not, then $c_r(\widetilde{P}) \geq 0$ for every st-dipath $\widetilde{P} \in \mathcal{P}_{st}$. Since $\lambda_a^{st} \geq 0$, for all $a \in \widetilde{A}_{st}$, the satellite problem can be solved in polynomial time. As the graphs $\widetilde{G}_{st}$ are circuitless, the shortest paths between s and t can be computed using for instance Bellman algorithm [11].

If $c_r(\widetilde{P}) \geq 0$ for all $\widetilde{P} \in \mathcal{P}_{st}$, $\{s,t\} \in D$, then the optimal solution of the current linear program is optimal for the linear relaxation of Formulation (5.20).

The initial sets $\mathcal{B}_{st}$ are chosen in the following way. For all $\{s,t\} \in D$, we add in $\mathcal{B}_{st}$ k st-dipaths of the form (s,t) or (s,u,u',t). To improve the convergence of the column generation algorithm, at each iteration we add to a set $\mathcal{B}_{st}$ all the dipaths of $\widetilde{G}_{st}$ having a negative reduced cost, that is having length $< -\lambda_0^{st}$. This can be done in polynomial time using Epstein [46] or Hershberger et al. algorithms [70]. For our purpose, we devise an algorithm which relies on the layered structure of the graph $\widetilde{G}_{st}$. The algorithm works as follows for a pair $\{s,t\} \in D$. First, we compute, using Bellman algorithm [11], the shortest paths from s to every other node of $\widetilde{V}_{st} \setminus \{s\}$, and let $l_{st}(u)$ denote the length of the shortest path from s to u, $u \in \widetilde{V}_{st} \setminus \{s\}$. If $l_{st}(t) \geq -\lambda_0^{st}$, then, for every st-dipath $\widetilde{P} \in \mathcal{P}_{st}$, $c_r(\widetilde{P}) \geq 0$. If $l_{st}(t) < -\lambda_0^{st}$, then at least one st-dipath will be added to $\mathcal{B}_{st}$. We first look for a path (s,t). If $\lambda_{(s,t)}^{st} < -\lambda_0^{st}$, then we add the path (s,t) to $\mathcal{B}_{st}$. Afterwards, we look for a st-dipath of the form (s,u,v',t), with $u \in N_{st}$ and $v' \in N'_{st}$. In fact, every st-dipath of $\widetilde{G}_{st}$ different from (s,t) is of the form (s,u,v',t). For every node $v' \in N'_{st}$, if $l_{st}(v') + \lambda_{(v',t)}^{st} < -\lambda_0^{st}$, then we add the st-path (s,u,v',t) to $\mathcal{B}_{st}$. We repeat this procedure for every $\{s,t\} \in D$. The algorithm is exact and runs in polynomial time.

6.2.2 Branch-and-Cut-and-Price algorithm

The optimal solution of the linear relaxation of Formulation (5.20) is feasible for Formulation (5.20) if it is integral. If this is not the case, then we add further valid

inequalities for $k\text{HNDP}_{PA}(G, D)$ that are violated by this solution. The inequalities that are considered are the following, in this order,

1. double cut inequalities,
2. triple path-cut inequalities,
3. Steiner-partition inequalities,
4. Steiner-SP-partition inequalities.

For our different Branch-and-Cut and Branch-and-Cut-and-Price algorithms, all the inequalities that are considered are global, that is valid for all the Branch-and-Cut tree, and several inequalities may be added at each iteration of the Branch-and-Cut and Branch-and-Cut-and-Price algorithms. These inequalities are lifted before their introduction in the current LP. We go to the next class of inequalities only if we have not found any violated inequality in the current class.

In the following section, we describe the different procedures we use to detect the violated inequalities.

6.3 Separation procedures

6.3.1 Separation of st-dicut inequalities

The separation of st-dicut inequalities (5.6) and (5.21) can be performed in polynomial time by computing, for every $\{s, t\} \in D$, a minimum weight st-dicut in $\widetilde{G}_{st}(\overline{y}_{st})$ (resp. $\widetilde{G}(\overline{y})$) with weights $(\overline{y}_{st}(a), a \in \widetilde{A}_{st}(\overline{y}_{st}))$ (resp. $(\overline{y}(a), a \in \widetilde{A}(\overline{y}))$) for inequalities (5.6) (resp. (5.21)). By minimum cut - maximum flow relationship, computing a minimum weight st-dicut of $\widetilde{G}_{st}(\overline{y}_{st})$ (resp. $\widetilde{G}(\overline{y})$) is equivalent to computing a maximum flow separating s and t. We use, for computing maximum flows, the efficient algorithm of Goldberg and Tarjan [58] which runs in $O(|\widetilde{V}_{st}||\widetilde{A}_{st}| \log \frac{|\widetilde{V}_{st}|^2}{|\widetilde{A}_{st}|})$, for all $\{s, t\} \in D$ (resp. $O(|\widetilde{V}||\widetilde{A}| \log \frac{|\widetilde{V}|^2}{|\widetilde{A}|})$). As this operation is repeated $|D|$ times, the whole algorithm runs in $O(|D||\widetilde{V}_{st}||\widetilde{A}_{st}| \log \frac{|\widetilde{V}_{st}|^2}{|\widetilde{A}_{st}|})$, for all $\{s, t\} \in D$ (resp. $O(|D||\widetilde{V}||\widetilde{A}| \log \frac{|\widetilde{V}|^2}{|\widetilde{A}|})$), and hence is polynomial time.

6.3.2 Separation of aggregated cut inequalities

To separate the aggregated cut inequalities, we consider the inequalities of type (5.29) and (5.33) and devise an heuristic to separate them. In particular, we consider the inequalities described in the following two lemmas. The separation procedure relies on a special graph (introduced later) defined with respect to $\widetilde{G}$ ($\widetilde{G}_{st}$, $\{s,t\} \in D$) and a fractional solution. Recall that these inequalities are valid for the polytopes kHNDP$_{Ag}(G,D)$ and kHNDP$_{Cu}(G,D)$.

Lemma 6.3.1 *Consider an inequality $\alpha x+\beta y \geq \gamma$ of type (5.29) induced by a node set family $\Pi = \{\widetilde{W}_1, ..., \widetilde{W}_p\}$, $p \geq 2$, and arc subsets $\widetilde{F}_i^0 \subseteq \delta^+_{\widetilde{G}}(\widetilde{W}_i)$ such that $|\widetilde{F}_i^0| = k-1$. Let $\widetilde{F} = \bigcup_{i=1}^{p}(\delta^+_{\widetilde{G}}(\widetilde{W}_i) \setminus \widetilde{F}_i^0)$, $\widetilde{F}_2$ be the set of arcs of $\widetilde{A}$ which appear twice in $\widetilde{F}$ and $\widetilde{F}_1$ those which appear once in $\widetilde{F}$. Suppose that for all arc $a \in \widetilde{F}_1$ there is another arc $a' \in \widetilde{F}_1$ which corresponds to the same edge of G as a. Let E_2 be the set of edges of G corresponding to the arcs of $\widetilde{F}_1$.*

If $(\overline{x},\overline{y}) \in \mathbb{R}^E \times \mathbb{R}^{\widetilde{A}}$ is a fractional solution of kHNDP$_{Ag}(G,D)$ such that $\overline{y}(\delta^+_{\widetilde{G}}(\widetilde{W}_i)) = k$ and $\overline{y}(a) = 1$, for all $a \in \widetilde{F}_i^0$, $i = 1, ..., p$, then $\alpha x + \beta y \geq \delta$ is violated by $(\overline{x},\overline{y})$ if and only if

$$2\sum_{e\in E_2}\overline{x}(e) - \sum_{a\in\widetilde{F}_1}\overline{y}(a) < 1. \tag{6.5}$$

Proof. First observe that inequality $\alpha x + \beta y \geq \delta$ is violated by $(\overline{x},\overline{y})$ if and only if

$$\sum_{a\in\widetilde{F}_2}\overline{y}(a) + \sum_{e\in E_2}\overline{x}(e) < \frac{p+1}{2}. \tag{6.6}$$

Since $\overline{y}(\delta^+_{\widetilde{G}}(\widetilde{W}_i)) = k$, $|\widetilde{F}_i^0| = k-1$ and $\overline{y}(a) = 1$ for all $a \in \widetilde{F}_i^0$, we have that

$$\overline{y}(\delta^+_{\widetilde{G}}(\widetilde{W}_i) \setminus \widetilde{F}_i^0) = 1 \text{ for } i = 1, ..., p.$$

Thus, $\sum_{i=1}^{p}\overline{y}(\delta^+_{\widetilde{G}}(\widetilde{W}_i) \setminus \widetilde{F}_i^0) = 2\sum_{a\in\widetilde{F}_2}\overline{y}(a) + \sum_{a\in\widetilde{F}_1}\overline{y}(a) = p$ and hence,

$$\sum_{a\in\widetilde{F}_2}\overline{y}(a) = \frac{p}{2} - \frac{1}{2}\sum_{a\in\widetilde{F}_1}\overline{y}(a). \tag{6.7}$$

From (6.6) and (6.7), we get

$$p - \sum_{a \in \widetilde{F}_1} \overline{y}(a) + 2 \sum_{e \in E_2} \overline{x}(e) < p + 1.$$

and the result follows. □

Lemma 6.3.2 *Consider an inequality* $\alpha x + \sum_{\{s,t\} \in D} y_{st}\beta_{st} \geq \gamma$ *of type (5.33) induced by a family of node sets* $\Pi = \{\widetilde{W}_1^{s_1t_1}, ..., \widetilde{W}_{p_1}^{s_1t_1}, ..., \widetilde{W}_1^{s_qt_q}, ..., \widetilde{W}_{p_q}^{s_qt_q}\}$, *with* $p_i \geq 1$, *for* $i = 1, ..., q$, *and* $p = \sum_{i=1}^{q} p_i \geq 2$, *and arc subsets* $\widetilde{F}_j^{s_it_i,0} \subseteq \delta^+_{\widetilde{G}_{s_it_i}}(\widetilde{W}_j^{s_it_i})$ *such that* $|\widetilde{F}_j^{s_it_i,0}| = k - 1$, $j = 1, ..., p_i$, $i = 1, ..., q$. *Let* $\widetilde{F}^{s_it_i} = \bigcup_{j=1}^{p_i} [\delta^+_{\widetilde{G}_{s_it_i}}(\widetilde{W}_j^{s_it_i}) \setminus \widetilde{F}_j^{s_it_i,0}]$, $i = 1, ..., q$. *Also let* $\widetilde{F}^{s_it_i,2}$ *be the set of arcs of* $\widetilde{A}_{s_it_i}$ *which appear twice in* $\widetilde{F}^{s_it_i}$ *and* $\widetilde{F}^{s_it_i,1}$ *those which appear once in* $\widetilde{F}^{s_it_i}$. *Suppose that for all arc* $a \in \widetilde{F}^{s_it_i,1}$, *there exists a unique arc* $a' \in \widetilde{F}^{s_{i'}t_{i'},1}$ *for some* $i' \in \{1, ..., q\}$ *which corresponds to the same edge of* G *as* a. *Let* E_2 *be the set of edges of* G *corresponding to these arcs.*

If $(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$ *is a fractional solution of* $kHNDP_{Cu}(G, D)$ *such that* $\overline{y}_{s_it_i}(\delta^+_{\widetilde{G}_{s_it_i}}(\widetilde{W}_{s_it_i})) = k$ *and* $\overline{y}_{s_it_i}(a) = 1$, *for all* $a \in \widetilde{F}^{s_it_i,0}$, $i = 1, ..., q$, *then* $\alpha x + \sum_{\{s,t\} \in D} y_{st}\beta_{st} \geq \gamma$ *is violated by* $(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$ *if and only if*

$$2 \sum_{e \in E_2} \overline{x}(e) - \sum_{i=1}^{q} \sum_{a \in \widetilde{F}^{s_it_i}} \overline{y}_{s_it_i}(a) < 1. \tag{6.8}$$

Proof. Similar to the proof of Lemma 6.3.1. □

In the following, we are going to discuss the separation of the aggregated cut inequalities (5.29) for kHNDP$_{Ag}$. After that, we will describe the separation procedure for the aggregated cut inequalities (5.33) related to kHNDP$_{Cu}$.

We are going to introduce an undirected graph, denoted by $H(\overline{x}, \overline{y})$, obtained from $\widetilde{G}$ and defined with respect to $(\overline{x}, \overline{y})$. As we will see in the following, the main property of this graph is that there is a matching between some particular cycles of $H(\overline{x}, \overline{y})$ and inequalities of type (5.29), described as in Lemma 6.3.1. The graph $H(\overline{x}, \overline{y})$ is obtained as follows.

For each arc of $\widetilde{A}$ having a fractional value with respect to $\overline{y}$, we add a node in $H(\overline{x},\overline{y})$. For convenience, we will denote by a the node of $H(\overline{x},\overline{y})$ corresponding to an arc a of $\widetilde{G}$. We add an edge in $H(\overline{x},\overline{y})$ between two nodes a_1 and a_2 if one of the conditions below is satisfied.

1. There exists an st-dicut of $\widetilde{G}(\overline{y})$, say $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$, for some $\{s,t\} \in D$, which contains a_1 and a_2, and such that $\overline{y}(\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})) = k$, $|\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \cap \widetilde{A}_1(\overline{y})| = k-1$ and $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \cap \widetilde{A}_f(\overline{y}) = \{a_1, a_2\}$.
2. The arcs a_1 and a_2 correspond to the same edge of G.

The edges added by Condition 1 will be said of *type 1* and those added by Condition 2 will be said of *type 2*. Figures 6.1 and 6.2 give respectively the support graph $\widetilde{G}(\overline{y})$ of a fractional solution $(\overline{x},\overline{y})$ of kHNDP$_{Ag}(G,D)$ and the graph $H(\overline{x},\overline{y})$ associated with that solution.

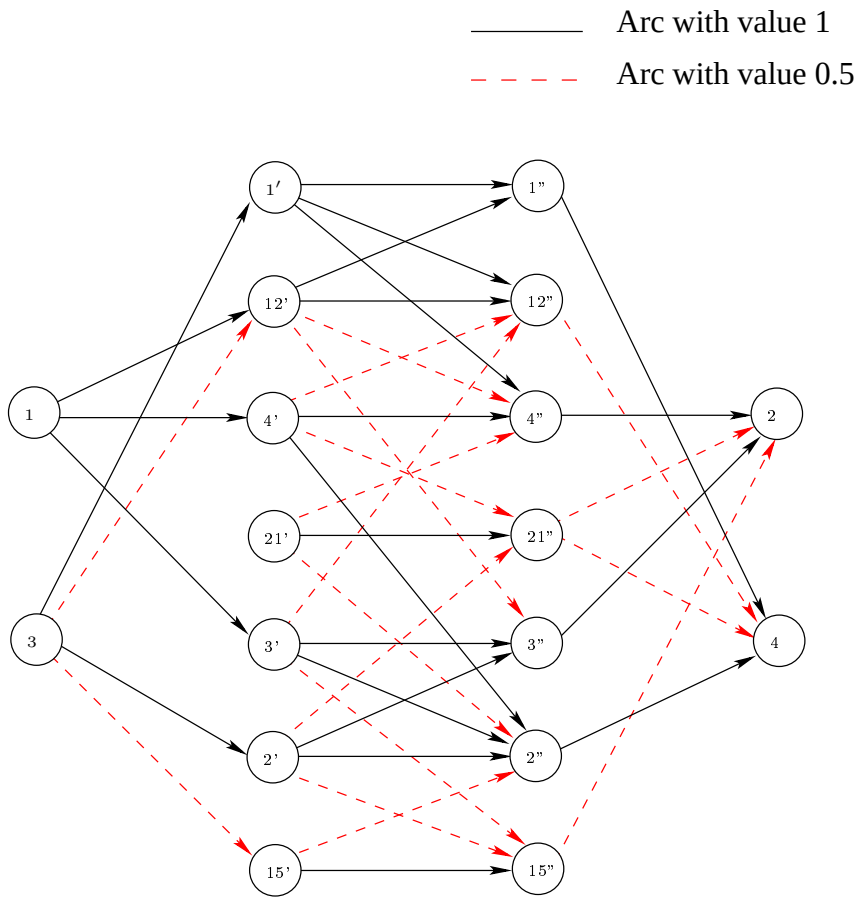


Figure 6.1: The support graph $\widetilde{G}(\overline{y})$ of a fractional solution $(\overline{x}, \overline{y})$ for $L = 3$ and $k = 3$

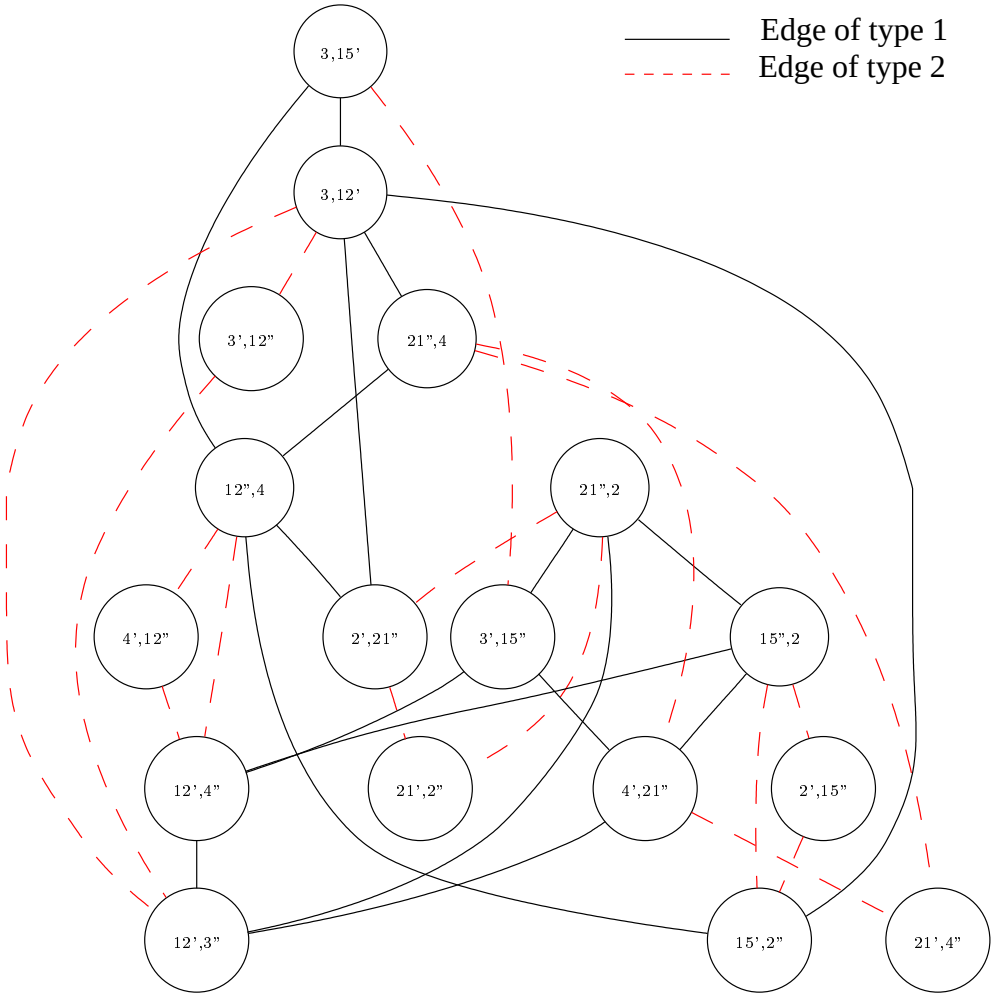


Figure 6.2: Graph $H(\overline{x}, \overline{y})$ obtained from $\widetilde{G}(\overline{y})$

Note that in the case where there is an edge of type 1 in $H(\overline{x},\overline{y})$ between two nodes a_1 and a_2, we have that $\overline{y}(a_1)+\overline{y}(a_2)=1$. Also, if there is an edge of type 2 between two nodes a_1 and a_2, then $\overline{x}(e)>0$ where e is the edge of G corresponding to a_1 and a_2. Also it is not hard to see that, if in $H(\overline{x},\overline{y})$ there are two edges of type 2 of the form a_1a_2 and a_2a_3, then there is also an edge of type 2 between a_1 and a_3 (a_1, a_2 and a_3 form a triangle).

Now we give the main property of $H(\overline{x},\overline{y})$.

Lemma 6.3.3 *Let $C=\{a_1a_2,a_2a_3,...,a_{|C|}a_1\}$ be a cycle of $H(\overline{x},\overline{y})$ and $\{a_{i_1}a_{j_1},...,a_{i_p}a_{j_p}\}$ the set of edges of C of type 1. Also, let V_1 be the set of nodes of C incident to two consecutive edges of type 1. Suppose that $p\geq 2$ and that C does not contain two consecutive edges of type 2. Then, C yields an inequality of type (5.29) defined by $\Pi=\{\widetilde{W}_1,...,\widetilde{W}_p\}$ and $\widetilde{F}_r^0=\delta_{\widetilde{G}(\overline{y})}(\widetilde{W}_r)\setminus\{a_{i_r},a_{j_r}\}$, $r=1,...,p$, where $\widetilde{W}_r$ is the node set of $\widetilde{G}$ associated with the edge $a_{i_r}a_{j_r}$ in the construction of $H(\overline{x},\overline{y})$.*

Proof. First observe that the arcs of $\widetilde{A}(\overline{y})$ which appear twice in $\widetilde{F}=\bigcup_{i=1}^{p}[\delta_{\widetilde{G}(\overline{y})}(\widetilde{W}_r)\setminus\widetilde{F}_i^0]$ are those of $\widetilde{G}(\overline{y})$ corresponding to the nodes of V_1, while the arcs which appear once in $\widetilde{F}$ are those of $\widetilde{A}(\overline{y})$ corresponding to the nodes of $\{a_1,...,a_{|C|}\}\setminus V_1$. Thus we let $\widetilde{F}_2$ and $\widetilde{F}_1$ be these two sets of arcs, respectively. Since every node $a\in\{a_1,...,a_{|C|}\}\setminus V_1$ is incident to one edge of C of type 2, say aa', the arcs a and a' are in $\widetilde{F}_1$ and correspond to the same edge of G. Thus, the aggregated cut inequality associated with this configuration can be written as

$$\sum_{a\in\widetilde{F}_2}y(a)+\sum_{e\in E_2}x(e)\geq\left\lceil\frac{p}{2}\right\rceil,$$

where E_2 is the edge set of G corresponding to the arcs of $\widetilde{F}_1$. □

To illustrate that lemma, on Figure 6.2, the cycle

$$C=\{(3,15')(3,12'),(3,12')(21'',4),(21'',4)(4',21''),(4',21'')(3',15''),(3',15'')(3,15')\}$$

contains three edges of type 1, $(3,15')(3,12')$, $(3,12')(21'',4)$ and $(4',21'')(3',15'')$, and two edges of type 2, $(21'',4)(4',21'')$ and $(3',15'')(3,15')$, that are not incident. One can see on Figure 6.1 that the node sets $\widetilde{W}_1=\{3\}$, $\widetilde{W}_2=\{3,2',15',21'',3'',2'',15'',2\}$ and $\widetilde{W}_3=\{1,12',3',4',1'',12'',4'',3'',2'',4\}$ induce two $3-4$-dicuts and one $1-2$-dicut of $\widetilde{G}(\overline{y})$, and that these dicuts contain respectively the pairs of arcs $\{(3,15'),(3,12')\}$,

$\{(3,12'),(21'',4)\}$ and $\{(4',21''),(3',15'')\}$. Moreover, they are such that $\overline{y}(\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}_i)) = k$ and $|\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}_i) \cap \widetilde{A}_1(\overline{y})| = k-1$, $i \in \{1,2,3\}$. Finally, it obviously follows that $\Pi = \{\widetilde{W}_1, \widetilde{W}_2, \widetilde{W}_3\}$ and $\widetilde{F}^0_1 = \{(3,1'),(3,2')\}$, $\widetilde{F}^0_2 = \{(3,1'),(2'',4)\}$ and $\widetilde{F}^0_3 = \{(4'',2),(3'',2)\}$ induce an aggregated cut inequality of type (5.29). Furthermore, this inequality is violated by $(\overline{x},\overline{y})$.

Before describing the construction procedure for $H(\overline{x},\overline{y})$, we give the following lemma.

Lemma 6.3.4 *Let $(\overline{x},\overline{y})$ be a fractional solution of $kHNDP_{Ag}(G,D)$, and let a_1 and a_2 be two arcs of $\widetilde{G}$ with fractional values and $\{s,t\} \in D$. If there exists a minimum weight st-dicut of $\widetilde{G}(\overline{y})$, say $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$, such that $\{a_1,a_2\} \subseteq \delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$ and $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \setminus \{a_1,a_2\} \subseteq \widetilde{A}_1(\overline{y})$, then $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$ can be considered in such a way that every arc $a \in \delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \setminus \{a_1,a_2\}$ is either in $\delta^+_{\widetilde{G}(\overline{y})}(s)$ or in $\delta^-_{\widetilde{G}(\overline{y})}(t) \setminus [t',t]_{\widetilde{G}(\overline{y})}$.*

Proof. Let $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$ be a minimum weight *st*-dicut of $\widetilde{G}(\overline{y})$ containing a_1 and a_2 and such that $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \setminus \{a_1,a_2\} \subseteq \widetilde{A}_1(\overline{y})$. Suppose also that there is an arc $a \in \delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \setminus \{a_1,a_2\}$ which is not in $\delta^+_{\widetilde{G}(\overline{y})}(s) \cup [\delta^-_{\widetilde{G}(\overline{y})}(t) \setminus \{(t',t)\}]$. Hence, a is either of the form (u',v''), with $u' \in N'$, $v'' \in N''$ and u and v may be the same, or of the form (t',t). If $a = (u',v'')$, then $u' \in \widetilde{W}$ and the node set $\widetilde{W}' = \widetilde{W} \setminus \{u'\}$ induces an *st*-dicut. Since $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$ is a minimum weight *st*-dicut, $[s,u']_{\widetilde{G}(\overline{y})} \neq \emptyset$ and therefore, $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}') = (\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \setminus \{(u',v'')\}) \cup \{(s,u')\}$. Since $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$ is of minimum weight with respect to $\overline{y}$, we have that $\overline{y}(s,u') \geq \overline{y}(u',v'')$. As $\overline{y}(u',v'') = 1$, we also have that $\overline{y}(s,u') = 1$ and that $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}')$ is a minimum weight *st*-dicut. If $a = (t',t)$, then since $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$ is of minimum weight in $\widetilde{G}(\overline{y})$, there is an arc of the form (s,t'). Thus, $\widetilde{W}' = \widetilde{W} \setminus \{t'\}$ induces an *st*-dicut of $\widetilde{G}(\overline{y})$. Moreover, as the weight of $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$ is minimum with respect to $\overline{y}$, we have that $\overline{y}(s,t') \geq \overline{y}(t',t) = 1$. Hence, $\overline{y}(s,t') = 1$ and $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}')$ is also of minimum weight.

By repeating this operation until $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$ does not contain any arc of the form (u',v'') or (t',t), we obtain a minimum weight *st*-dicut of $\widetilde{G}(\overline{y})$ which contains a_1 and a_2, such that $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \setminus \{a_1,a_2\} \subseteq \widetilde{A}_1(\overline{y})$ and such that every arc of $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \setminus \{a_1,a_2\}$ is either in $\delta^+_{\widetilde{G}(\overline{y})}(s)$ or in $\delta^-_{\widetilde{G}(\overline{y})}(t) \setminus [t',t]_{\widetilde{G}(\overline{y})}$, which ends the proof of the Lemma. □

A consequence of Lemma 6.3.4 is that an *st*-dicut $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$ of $\widetilde{G}(\overline{y})$ containing two arcs a_1 and a_2 with fractional values, such that $\overline{y}(\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})) = k$ and $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \cap \widetilde{A}_f(\overline{y}) =$

$\{a_1, a_2\}$ can be obtained by computing st-dicuts of $\widetilde{G}(\overline{y})$ containing a_1 and a_2 and such that $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \setminus \{a_1, a_2\} \subseteq \left[\delta^+_{\widetilde{G}(\overline{y})}(s) \cup (\delta^-_{\widetilde{G}(\overline{y})}(t) \setminus \{(t', t))\}\right]$.

The construction of the graph $H(\overline{x}, \overline{y})$ is performed by computing first the edges of type 2. For every pair of arcs $(a, a') \in \widetilde{A}(\overline{y}) \times \widetilde{A}(\overline{y})$, corresponding to the same edge of E and having a fractional value, we add an edge of type 2 between the corresponding nodes in $H(\overline{x}, \overline{y})$. To compute the edges of type 1, we use a procedure based on Lemma 6.3.4. The idea is to compute a maximum flow in $\widetilde{G}(\overline{y})$ with respect to appropriate capacities separating s and t. Given two arcs a_1 and a_2 such that $\overline{y}(a_1) + \overline{y}(a_2) = 1$ and a pair $\{s, t\} \in D$, we first give 0 as capacity to a_1 and a_2. Then, we give an infinit capacity to every other arc of $\widetilde{G}(\overline{y})$ having a fractional value. This ensures that a_1 and a_2 are the only arcs of fractional values present in the st-dicut we will obtain. We give an infinit capacity to every arc of $\delta^+_{\widetilde{G}(\overline{y})}(s)$ and $\delta^-_{\widetilde{G}(\overline{y})}(t)$ indicent to a_1 and a_2 and having value 1. We also give an infinit capacity to every arc of $[t', t]_{\widetilde{G}(\overline{y})}$. For all other arc a, we give $\overline{y}(a)$ as capacity (note that for these arcs, $\overline{y}(a) = 1$). Then, we compute a maximum flow between s and t with respect to these capacities. Let $\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})$ denote the st-dicut thus obtained. By Lemma 6.3.4, we have that $\delta^+_{\widetilde{G}(\overline{y})} \setminus \{a_1, a_2\} \subseteq \widetilde{A}_1(\overline{y})$. We then check if $\overline{y}(\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W})) = k$ and $|\delta^+_{\widetilde{G}(\overline{y})}(\widetilde{W}) \setminus \{a_1, a_2\}| = k - 1$. If this is the case, then we add an edge of type 1 between the nodes of $H(\overline{x}, \overline{y})$ corresponding to a_1 and a_2. We repeat this procedure for all pair of arcs (a_1, a_2) having fractional value and such that $\overline{y}(a_1) + \overline{y}(a_2) = 1$, and for all demand $\{s, t\} \in D$.

Now we describe the separation procedure of the aggregated cut inequalities. The procedure is based on Lemma 6.3.1. Thus we generate inequalities of type (5.29) which satisfy the conditions of that lemma. First, we compute $H(\overline{x}, \overline{y})$ as described above. Then we compute one or more cycles of $H(\overline{x}, \overline{y})$ which contain an odd number of edges of type 1 and which does not contain two consecutive edges of type 2. By Lemma 6.3.3, every cycle satisfying these conditions yields an aggregated cut inequality of type (5.29). We then check if for each inequality thus obtained, $(\overline{x}, \overline{y})$ satisfies inequality (6.5). If this is the case, then by Lemma 6.3.1, this inequality is violated by $(\overline{x}, \overline{y})$ and added to the set of violated inequalities. If no cycle is found or if for every inequality of type (5.29) obtained, $(\overline{x}, \overline{y})$ does not satisfy inequality (6.5), then the procedure ends with failure.

To detect cycles of $H(\overline{x}, \overline{y})$ satisfying the conditions of Lemma 6.3.3, we use a procedure in which we compute shortest paths in an auxiliary graph obtained from $H(\overline{x}, \overline{y})$. Let H_b be the undirected graph obtained as follows. The node set of H_b is composed of two copies, denoted by V'_b and V''_b, of the node set of $H(\overline{x}, \overline{y})$. The copies of a node a of $H(\overline{x}, \overline{y})$ are denoted by a' and a'' with $a' \in V'_b$ and $a'' \in V''_b$. For every edge a_1a_2

of $H(\overline{x}, \overline{y})$ of type 1, we add in H_b two edges of the form $a'_1 a''_2$ and $a'_2 a''_1$ and give them 1 as length. For every edge $a_1 a_2$ of $H(\overline{x}, \overline{y})$ of type 2, we add in H_b two edges of the form $a'_1 a'_2$ and $a''_1 a''_2$ and give them a length M sufficiently large. Figure 6.3 shows an example of graph H_b obtained from a subgraph of $H(\overline{x}, \overline{y})$ given in Figure 6.2. It is not hard to see that a path between two nodes a' and a'' of H_b corresponds to a cycle of $H(\overline{x}, \overline{y})$ containing node a and an odd number of edges of type 1, and does not contain two consecutive edges of type 2, and vice versa.

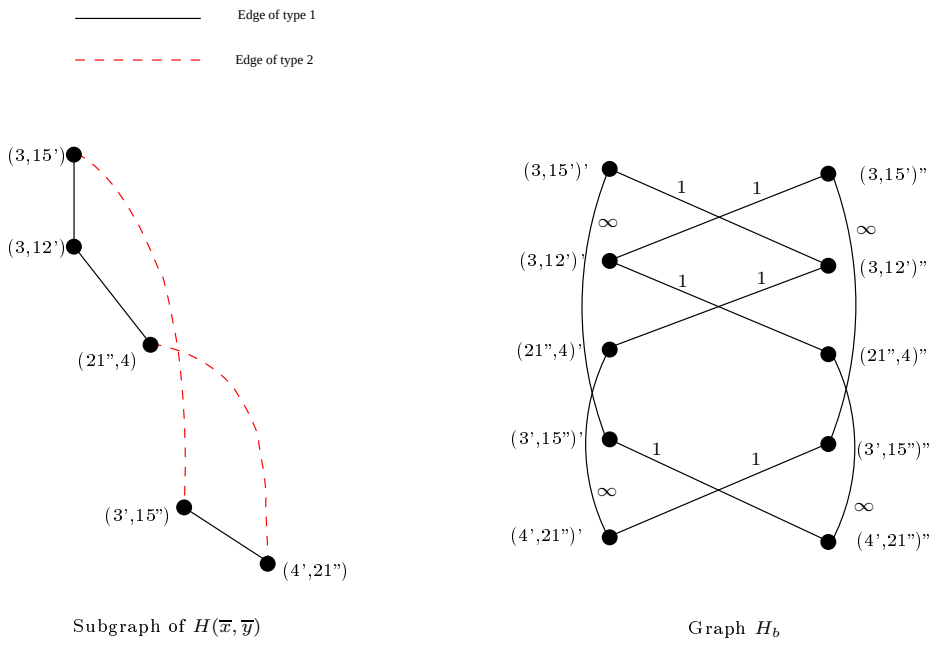


Figure 6.3: Graph H_b obtained from a subgraph of $H(\overline{x}, \overline{y})$

For our separation procedure, we compute the shortest paths between each pair of nodes (a', a'') of H_b, for every node a of $H(\overline{x}, \overline{y})$.

Now we turn to the aggregated cut inequalities for the Cut formulation. The separation procedure for these inequalities is similar to that described above for kHNDP$_{Ag}$. Given a fractional solution $(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$ of kHNDP$_{Cu}(G, D)$, we construct the graph $H(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$ in a similar way as $H(\overline{x}, \overline{y})$, that is for all $\{s, t\} \in D$, and for every arc $a \in \widetilde{A}^f_{st}(\overline{y}_{st})$ we associate a node in $H(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$. We add an edge, said of type 1, between two nodes a_1 and a_2 if they belong to the same graph $\widetilde{G}_{st}$, $\overline{y}_{st}(a_1) + \overline{y}_{st}(a_2) = 1$ and there exists an st-dicut $\delta^+_{\widetilde{G}_{st}(\overline{y}_{st})}(\widetilde{W})$ containing a_1 and a_2 and such that $\delta^+_{\widetilde{G}_{st}(\overline{y}_{st})}(\widetilde{W}) \cap \widetilde{A}^f_{st}(\overline{y}_{st}) = \{a_1, a_2\}$ and $|(\delta^+_{\widetilde{G}_{st}(\overline{y}_{st})}(\widetilde{W}) \setminus \{a_1, a_2\}) \cap \widetilde{A}^1_{st}(\overline{y}_{st})| = k - 1$. We also add an edge, said of type 2, between two nodes $a_1 \in \widetilde{A}^f_{s_it_i}(\overline{y}_{s_it_i})$ and $a_2 \in \widetilde{A}^f_{s_{i'}t_{i'}}(\overline{y}_{s_{i'}t_{i'}})$ if the arcs a_1 and a_2 correspond to the same edge of G.

The st-dicut $\delta^+_{\widetilde{G}_{st}(\overline{y}_{st})}(\widetilde{W})$ used to set edges of type 1 can be computed with the procedure used for kHNDP$_{Ag}$. As before, every cycle of $H(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$ which contains an odd number of edges of type 1 and which does not contain two consecutive edges of type 2 yields an inequality of type (5.33). These cycles are computed by looking for shortest paths in a graph H_b obtained in a similar way as for kHNDP$_{Ag}$. Finally, for each cycle thus obtained, we check if $(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$ satisfies or not inequality (6.8) with respect to the sets E_2 and $\widetilde{F}^{s_it_i,1}$ obtained from that cycle. If this is the case, then by Lemma 6.3.2, the corresponding inequality of type (5.33) is violated by $(\overline{x}, \overline{y}_{s_1t_1}, ..., \overline{y}_{s_dt_d})$ and hence added to the set of violated inequalities.

6.3.3 Separation of double cut inequalities

The separation of double cut inequalities is performed by looking for inequalities of type (5.39) for $L = 2$ and of type (5.40) for $L = 3$ that are violated by the current solution. We describe the procedure for the kHNDP$_{Ag}$. We will present later how this can be extended to the other formulations.

The idea of the procedure is to find a partition $\pi = (V_0, ..., V_L, V_{L+1})$, $L \in \{2, 3\}$, of G and an edge set $F \subseteq E$, with $|V_0| = |V_1| = 1$ and $[V_0, V_1] \neq \emptyset$, which induces a double cut, with $i_0 = 0$, and whose weight is minimum with respect to $\overline{x}$. The procedure works as follows. For all $\{s, t\} \in D$, we compute the st-cut $\delta_G(s)$. If $\overline{x}(\delta_G(s)) = k$, then for every terminal $s' \in R_D$ such that $\overline{x}([s, s']) > 0$ and $\overline{x}(\delta_G(s')) = k$, we compute an L-st-path-cut T of G induced by a partition $\pi = (V_0, ..., V_L, V_{L+1})$ with $V_0 = \{s\}$ and $V_1 = \{s'\}$. For this, we use the correspondance between L-st-path-cuts in G and st-dicuts in $\widetilde{G}$, given by Lemma 5.4.1. Since the desired partition π must be such that $V_0 = \{s\}$ and $V_1 = \{s'\}$, we must have $T \cap [s, s'] = \emptyset$ and $\delta_G(s) \setminus [s, s'] \subseteq T$. Thus, any st-dicut of $\widetilde{G}$ corresponding to T must contain arcs corresponding to the edges of $\delta_G(s) \setminus [s, s']$ and no arcs corresponding to the edges of $[s, s']$. Also remark that this st-dicut does not contain any arc of the form (u', u''), $u \in V$ and of the form (t', t), $t \in T_D$. Therefore, to compute an st-dicut of $\widetilde{G}$ corresponding to the desired L-st-path-cut, we start by giving the arcs corresponding to the edges of $[s, s']$ an infinit capacity and removing all the arcs corresponding to the edges of $\delta_G(s) \setminus [s, s']$. Then, we give to every arc of the form (u', u''), $u \in V$ and (t', t), $t \in T_D$, an infinit capacity. Afterwards, we compute a maximum flow between s and t with respect to these capacities. Let $\delta^+_{\widetilde{G}}(\widetilde{W})$ denote the st-dicut thus obtained.

To check that this dicut corresponds to an L-st-path-cut of G, we apply the following procedure. We first remove from G all the edges corresponding to the arcs of $\delta^+_{\widetilde{G}}(\widetilde{W})$.

Then, we compute the shortest paths between s and every node of $V \setminus \{s\}$ with respect to length 1 on the remaining edges. Let $l(u)$ denotes the length of a shortest path between s and u, $u \in V \setminus \{s\}$. If $l(t)$ is finite, then $\delta^+_{\widetilde{G}}(\widetilde{W})$ corresponds to an L-st-path-cut of G. In this case, we construct the partition π such that $V_0 = \{s\}$, $V_1 = \{s'\}$, $V_i = \{u \in V \setminus \{s, s', t\} \mid l(u) = i\}$, $i = 2, ..., L$, and $V_{L+1} = V \setminus (\bigcup_{i=0}^{L} V_i)$.

Let $\hat{E}$ be the edge set $[V_1, V_2]$ (resp. $[V_1 \cup V_4, V_2]$) if $L = 2$ (resp. $L = 3$) having a positive value with respect to $\overline{x}$. We choose the edges of F among those of $\hat{E}$ having the highest value and such that $|F|$ and k have different parities. If $|\hat{E}| \geq k - 1$, then F consists of the $k - 1$ edges having the highest value. If $|\hat{E}| < k - 1$ and $|\hat{E}|$ has a parity different from that of k, then we let $F = \hat{E}$. If $|\hat{E}| < k-1$ and $|\hat{E}|$ has the same parity as k, then we let $F = \hat{E} \setminus \{e_0\}$ where e_0 is the edge of $\hat{E}$ having the smallest value.

Finally, we check if the inequality (5.39) (resp. (5.40)) for $L = 2$ (resp. $L = 3$) induced by π and F is violated or not.

We repeat this procedure for every demand $\{s, t\} \in D$, and the violated inequalities found are added to the constraint pool. To compute the maximum flow in $\widetilde{G}$ we use the algorithm of Goldberd and Tarjan [58] which runs in $O(|\widetilde{A}||\widetilde{V}| \log \frac{|\widetilde{V}|^2}{|\widetilde{A}|})$ time. If G is complete and $L = 3$, we have that $|\widetilde{V}| = 2|V| + |S_D| + |T_D|$ and $|\widetilde{A}| = (|V| - 1)(|V| + |S_D| + |T_D|)$. Thus, the maximum flow algorithm runs in $O(|V|^3 \log \frac{(2|V|+|S_D|+|T_D|)^2}{(|V|-1)(|V|+|S_D|+|T_D|)})$. To compute the shortest paths in G between s and the other nodes of V, we use the algorithm of Dijkstra [43] which is implemented to run is $O(|V||E| \log(|V|))$ time. As the computation of a cut in the graph G requires at most $|E|$ iterations, our separation procedure runs in $O(|V|^3 \log |V| \frac{(2|V|+|S_D|+|T_D|)^2}{(|V|-1)(|V|+|S_D|+|T_D|)}))$ time, and hence is polynomial. If $L = 2$, the algorithm is also polynomial.

For the case of the separated formulations (Cut, Path-Arc and Node-Arc formulations), the procedure is the same except that the computation of the L-st-path-cut, induced by the partition π, is performed using the directed graph $\widetilde{G}_{st}$ associated with the demand $\{s, t\}$. We remove from $\widetilde{G}_{st}$ all the arcs corresponding to the edges of $\delta_G(s) \setminus [s, s']$, and those corresponding to the edges of $[s, s']$ are given an infinit capacity. In the same way, we give an infinit capacity to every arc of the form (u, u'), with $u \in \widetilde{V}_{st}$. Then, we compute a maximum flow between s and t in $\widetilde{G}_{st}$. Also, for these formulations, the algorithm remains polynomial.

6.3.4 Separation of triple path-cut inequalities

To separate triple path-cut inequalities, we devise a heuristic. This heuristic is based on Theorem 5.7.3. The procedure is given for $L = 3$. It is similar for $L = 2$. The main idea is to compute, given two demands $\{s, t_1\}$ and $\{s, t_2\}$, a family $\Pi = \{V_0, V_1, V_2, V_3, V_4^1, V_4^2, V_5^1, V_5^2\}$ of node sets from a 3-st_1-path-cut T induced by a partition of the form $(V_0, V_1 \cup V_4^1 \cup V_4^2, V_2, V_3 \cup V_5^1, V_5^2)$. In fact, from this latter partition, one can obtain a whole triple path-cut by fixing the sets V_4^1, V_4^2, V_5^1 and V_5^2. In our procedure, we will look for those triple path-cuts such that $V_4^1 = \emptyset$, $V_4^2 = \{t_2\}$, $V_5^1 = \emptyset$ and $V_5^2 = \{t_1\}$.

The procedure works as follows. For each source $s \in S_D$, we apply the following steps. Let $\{s, t_1\}$ and $\{s, t_2\}$ be two demands associated with s. We first look for a partition $\pi = (V_0', V_1', V_2', V_3', V_4')$ which induces an L-st_1-path-cut of G, denoted by T, and such that $V_0' = \{s\}$ and $t_2 \in V_1'$. For this, we use the correspondance between the L-st_1-path-cuts in G and st_1-dicuts in $\widetilde{G}$. Since $t_2 \in V_1'$ and $V_0' = \{s\}$, we have that $T \cap [s, t_2] = \emptyset$ and any arc of $\widetilde{G}$, corresponding to the edges of $[s, t_2]$, does not appear in an st_1-dicut of $\widetilde{G}$ corresponding to T. Thus, computing T reduces to compute a minimum weight st_1-dicut in $\widetilde{G}$. To do this, we compute a maximum flow in $\widetilde{G}$ between s and t_1 with respect to the following capacities:

- for every arc of $\widetilde{A}([s, t_2])$ or of the form (u', u'') or (t', t), with $u \in N$ and $t \in T_D$, we give an infinit capacity;
- for every arc of $\widetilde{A}(e)$, with $e \in E \setminus [s, t_2]$, we give the capacity $\overline{x}(e)$.

Let $\delta^+_{\widetilde{G}}(\widetilde{W})$ denote the directed cut thus obtained. We check if it corresponds to an L-st_1-path-cut by performing the following steps. First, we remove from G all the edges corresponding to the arcs of $\delta^+_{\widetilde{G}}(\widetilde{W})$ and compute all the shortest paths between s and the other nodes of G with respect to the length 1 on the remaining edges. Let $l(u)$ denote the length of the shortest path between s and u, for all $u \in V \setminus \{s\}$. If $l(t_1)$ is finite, then $\delta^+_{\widetilde{G}}(\widetilde{W})$ corresponds to an L-st_1-path-cut, denoted by T. In this case, we construct the partition π such that $V_0' = \{s\}$, $V_i' = \{u \in V \mid l(u) = i\}$, for $i \in \{1, 2, 3\}$, and for all the nodes $u \in V \setminus \{t_1\}$ such that $l(u) \geq 4$ or $l(u) = +\infty$, we assign them alternatively to V_1' and V_3'. Finally, $V_4' = V \setminus (\bigcup_{i=0}^{3} V_i')$. Note that $t_1 \in V_4'$ as $l(t_1) > 3$ and $t_2 \in V_1'$. Now the family of node sets Π is such that $V_0 = V_0' = \{s\}$, $V_1 = V_1' \setminus \{t_2\}$, $V_2 = V_2', V_3 = V_3'$, $V_4^1 = \emptyset$, $V_4^2 = \{t_2\}$, $V_5^1 = \emptyset$ and $V_5^2 = \{t_1\}$.

Let $\hat{E}$ be the set of edges of $[V_3^2, V_1 \cup V_4^1] \cup [V_3^1, V_4^2]$ having a positive value with respect to $\overline{x}$. We choose the edges of F among those of $\hat{E}$ having the highest value and such that $|F|$ and k have different parities. If $|\hat{E}| \geq k-1$, then F consists of the $k-1$ edges having the highest value. If $|\hat{E}| < k-1$ and $|\hat{E}|$ has a parity different from that of k, then we let $F = \hat{E}$. If $|\hat{E}| < k-1$ and $|\hat{E}|$ has the same parity as k, then we let $F = \hat{E} \setminus \{e_0\}$ where e_0 is the edge of $\hat{E}$ having the smallest value.

Finally, we check if the triple path cut inquality induced by Π and F is violated or not.

Our algorithm runs in polynomial time, as it consists, for every pair $\{\{s,t_1\},\{s,t_2\}\}$ of demands, in computing a maximum flow and shortest paths between s and the other nodes of G. In our implementation, we use the algorithm of Goldberg and Tarjan [58] for the maximum flow and the algorithm of Dijkstra [43] for the shortest paths which run repesctively in $O(|V|^3 \log \frac{(2|V|+|S_D|+|T_D|)^2}{(|V|-1)(|V|+|S_D|+|T_D|)})$ and $O(|V|^3 \log |V|)$ time, respectively. Thus, the procedure runs in $O(|D|^2(|V|^3 \log |V| \frac{(2|V|+|S_D|+|T_D|)^2}{(|V|-1)(|V|+|S_D|+|T_D|)}))$ time, and thus, is polynomial.

For the case of the separated formulations, the procedure is the same except that the computation of the *L-st*-path-cut inducing the partition π is performed using the directed graph $\widetilde{G}_{st_1}$ associated with the demand $\{s,t_1\}$. All the arcs corresponding to the edges of $[s,t_2]$ are given an infinit capacity. In the same way, we give an infinit capacity to every arc of the form (u,u'), with $u \in \widetilde{V}_{st}$ and all the arcs corresponding to an edge $e \in E \setminus [s,t_2]$ is given the capacity $\overline{x}(e)$. Then, we compute a maximum flow between s and t_1 in $\widetilde{G}_{st_1}$.

6.3.5 Separation of Steiner-partition inequalities

Now we discuss the separation of Steiner-partition inequalities. The separation problem of inequalities (5.43) is NP-hard (see [99]). To separate them, we devise the following heuristic. Note that we look for Steiner-partition inequalities when k is odd. The idea of the procedure is to find a partition $\pi = (V_0, V_1, ..., V_p)$, $p \geq 3$ and odd, such that $V_0 \subseteq V \setminus R_D$ and $\overline{x}(\delta(V_0, ..., V_p))$ is minimum.

Our heuristic begins by contracting every pair of nodes t and u, where t is a terminal node and u a Steiner node, and $\overline{x}(\delta_{G(\overline{x})}(u) \setminus \{ut\}) \leq \overline{x}(ut)$. The node resulting from that contraction will considered as a terminal. Let $G(\overline{x})' = (V', E')$ be the reduced graph thus obtained and let $\{u'_1, ..., u'_p\}$ be the set of terminals of $G(\overline{x})'$. If p is odd, we let $\pi' = (V'_0, V'_1, ..., V'_p)$, where $V'_i = \{u'_i\}$, $i = 1, ..., p$, and $V'_0 = V' \setminus \{u'_1, ..., u'_p\}$.

Then, we let V_i, $i = 0, ..., p$, be the node sets of $G(\overline{x})$ corresponding to the node sets V'_i, $i = 0, ..., p$, of $G(\overline{x})'$.
If p is even, we look for two nodes u'_{i_0} and u'_{j_0}, $i_0, j_0 \in \{1, ..., p\}$, of $G(\overline{x})'$ such that $\overline{x}([u'_{i_0}, u'_{j_0}])$ is maximum and there is a demand between u'_{i_0} and u'_{j_0}, that is $|\delta_{G_D}(\{u'_{i_0}, u'_{j_0}\})| \geq 1$. This later condition ensures that the partition we will obtain is admissible. We let

$$\begin{aligned}
&V'_i = \{u'_i\},\ i = 1, ..., i_0 - 1,\\
&V'_{i_0} = \{u'_{i_0}, u'_{j_0}\},\\
&V'_i = \{u'_i\},\ i = i_0 + 1, ..., j_0 - 1,\\
&V'_{i-1} = \{u'_i\},\ i = j_0 + 1, ..., p,\\
&V'_0 = V' \setminus \{u'_1, ..., u'_p\}.
\end{aligned}$$

Then, we let V_i be the node set of $G(\overline{x})$ corresponding to the node set V'_i, $i = 0, ..., p-1$, of $G(\overline{x})'$. After that, we check if the Steiner-partition inequality induced by π is violated by $\overline{x}$ or not.

The computation of the graph $G(\overline{x})'$ runs in $O(|V||E|)$ while the computation of the nodes u'_{i_0}, u'_{j_0}, when p is even, requires $O(|R_D|^2(|E'| + |D|))$ operations. Thus, our separation algorithm runs, in the worst case, in $O(|V||E| + |R_D|^2(|E'| + |D|))$ time and thus, is polynomial.

6.3.6 Separation of Steiner-SP-partition inequalities

Now we turn our attention to the separation of the Steiner-SP-partition inequalities. We devise the following heuristic to separate inequalities (5.44). The main idea is to determine a Steiner-partition $\pi = (V_1, ..., V_p)$, $p \geq 3$, of V which induces an outerplanar subgraph of $G(\overline{x})$ and such that the subgraph of G_D (the demand graph) induced by π is connected. By Theorem 5.6.8, such a partition is a Steiner-SP-partition. Also, the partitions we are looking for are such that $|[V_i, V_{i+1}]| \geq \lceil \frac{k}{2} \rceil$, $i = 1, ..., p$, (modulo p) and for every consecutive sets V_i and V_j, the edge set $[V_i, V_j]$ contains at least one edge with fractional value.

The heuristic works as follows. We first contract every pair of nodes t and u, where t is a terminal node, u is a steiner node and $\overline{x}(\delta_{G(\overline{x})}(u) \setminus \{ut\}) \leq \overline{x}(ut)$. The node resulting from that contraction is said to be terminal. Let $G(\overline{x})' = (V', E')$ be the reduced graph thus obtained.

We look in $G(\overline{x})'$ for a path $\Gamma = \{v'_1v'_2, v'_2v'_3, ..., v'_{p-2}v'_{p-1}\}$, $p \geq 3$, such that $v'_1, ..., v'_{p-1}$ are terminal nodes, $|[v'_i, v'_{i+1}]| \geq \lceil \frac{k}{2} \rceil$ and $[v'_i, v'_{i+1}]$ contains one edge or more with frac-

tional value, for $i = 1, ..., p-2$. The partition $\pi = (V_1, ..., V_p)$, $p \geq 3$, is constructed such that V_i is the node set of G corresponding to v'_i, $i = 1, ..., p-1$, and $V_p = V \setminus (\bigcup_{i=1}^{p-1} V_i)$.

Afterwards, we check by a simple heuristic if the graph $G_\pi(\overline{x})'$ is outerplanar and if the subgraph of G_D induced by π is connected. If it is connected, then, we check if the Steiner-SP-partition inequality induced by π is violated. If this subgraph is not connected, we compute from π new partitions $\pi_i = (V_i, V_{i+1}, V \setminus (V_i \cup V_{i+1}))$, $i = 1, ..., p-2$. Clearly, these new partitions are Steiner-partitions and since they are of size 3, they induce Steiner-SP-partitions. We then check if the Steiner-SP-partition inequality induced by π_i is violated, for $i = 1, ..., p-2$.

If none of these inequalities is violated by $\overline{x}$, we apply again the procedure by looking for another path. In order to avoid the detection of the same path, we label the nodes we met during the search of the previous ones, so that they won't be considered in the search of the new path. This process is iterated until either we find a violated Steiner-SP-partition inequality or all the nodes of V' are labeled. The heuristic can be implemented to run in $O(|E'||V'| + |D|)$ time.

To store the generated inequalities, we create a pool whose size increases dynamically. All the generated inequalities are put in the pool and are dynamic, that is, they are removed from the current LP when they are not active. We first separate inequalities from the pool. If all the inequalities in the pool are satisfied by the current LP-solution, we separate the classes of inequalities in the order given before.

6.3.7 Primal heuristic

An important issue in the efficiency of the Branch-and-Cut and Branch-and-Cut-and-Price algorithms is to compute a good upper bound at each node of the Branch-and-Cut tree. To do this, when the separation procedures do not generate any violated inequality and the current solution is still fractional, we transform it into a feasible one. We describe the procedure we have devised for kHNDP$_{Ag}$ with a fractional solution $(\overline{x}, \overline{y})$. It is similar for kHNDP$_{Cu}$, kHNDP$_{PA}$ and kHNDP$_{NA}$. The main idea is to construct a graph obtained by removing from $\widetilde{G}(\overline{y})$ every arc corresponding to an edge of $G(\overline{x})$ having a fractional value and add arcs in that graph until the number of arc-disjoint st-dipaths reaches k, for all $\{s, t\} \in D$. Note that since $(\overline{x}, \overline{y})$ is fractional and is an optimal solution for the current LP, the restriction of $\widetilde{G}(\overline{y})$ to $\widetilde{A}_1(\overline{y})$ cannot contain k arc-disjoint st-dipaths for all $\{s, t\} \in D$. Otherwise, $(\overline{x}, \overline{y})$ would be integral or would not be optimal for the current LP.

The procedure relies on the computation of a maximum flow between s and t for every pair $\{s,t\} \in D$. The capacities of the arcs of $\widetilde{G}(\overline{y})$ are updated at each iteration of the procedure. At the end of the procedure, we remove from $\widetilde{G}(\overline{y})$ every arc whose capacity is null.

Let $C_i = (C_i(a))_{a \in \widetilde{A}(\overline{y})}$ be a capacity vector obtained at the end of the ith iteration, $i = 0, ..., d$, with $C_0 = (C_0(a))_{a \in \widetilde{A}(\overline{y})}$, where $C_0(a) = 1$ for all $a \in \widetilde{A}_1(\overline{y})$ and $C_0(a) = 0$ otherwise. Note that the capacity vector C_i, $i \in \{1, ..., d\}$, is associated with demand $\{s_i, t_i\}$. For a demand $\{s_i, t_i\}$, $i \in \{1, ..., d\}$, we first compute a maximum $s_i t_i$-flow with respect to capacity vector C_{i-1}. Let $f = (f(a))_{a \in \widetilde{A}(\overline{y})}$ be the corresponding flow vector and f_0 the value of this flow. If $f_0 \geq k$, then there is nothing to do for this demand. Thus we let $C_i(a) = C_{i-1}(a)$ for all $a \in \widetilde{A}(\overline{y})$ and go to the next demand $\{s_{i+1}, t_{i+1}\}$. If $f_0 < k$, then we compute $k - f_0$ arc-disjoint augmenting $s_i t_i$-paths with respect to capacity 1 on every arc of $\widetilde{G}(\overline{y})$ and $f(a)$ as initial flow value. Remark that the flow is null for all arc a having $C_{i-1}(a) = 0$. Then, we set to 1 the flow on every arc involved in the $k - f_0$ augmenting paths computed before and update the capacity vector C_i in the following way:

- $C_i(a) = 1$, for all $a \in \widetilde{A}(\overline{y})$ such that $C_{i-1}(a) = 1$;
- $C_i(a) = 1$, if $C_{i-1}(a) = 0$ and a is involved in an augmenting path computed before;
- $C_i(a) = 0$ otherwise.

We repeat this operation for every demand $\{s_i, t_i\}$, $i = 1, ..., d$. At the end of the procedure, we remove from $\widetilde{G}(\overline{y})$ every arc such that $C_d(a) = 0$. Afterwards, we construct the graph $\hat{G} = (V, \hat{E})$, where $\hat{E}$ is the set of edges associated with an arc remaing in $\widetilde{G}(\overline{y})$, that is having $C_d(a) = 1$. Since the remaining graph $\widetilde{G}(\overline{y})$ contains k arc-disjoint st-paths for all $\{s,t\} \in D$, the graph $\hat{G}$ contains k edge-disjoint L-st-paths, for all $\{s,t\} \in D$, and hence, induces a feasible solution of the kHNDP.

If the weight of this solution is lower than best known upper bound, then we update this upper bound with the weight of the solution we have just computed.

6.4 Computational results

The Branch-and-Cut and Branch-and-Cut-and-Price algorithms described in the previous sections have been implemented in C++, using ABACUS 3.0 [1, 101] to manage

the Branch-and-Cut tree, and CPLEX 11.0 [2] as LP-solver. It was tested using a machine equiped with a processor Intel Centrino Duo and 2 Go of RAM, running under Linux. The maximum CPU time has been fixed to 5 hours. The test problems we have considered are complete euclidian graphs from TSPLIB library [3]. The demands used in these tests are randomly generated. Each set of demand is either rooted in a node s, or is such that there is no demand having the same destination node as another demand. The tests have been performed for $L = 2, 3$ and $k = 3, 4, 5$.

Each instance is given by the number of nodes of the graph preceded by the type of demand, indicated by 'r' for rooted demands and 'a' for arbitrary demands. The other entries of the various tables are:

$\|V\|$	:	number of nodes of the graph;
$\|D\|$	:	number of demands,
NC	:	number of generated cut inequalities;
NAC	:	number of generated aggregated cut inequalities;
NDC	:	number of generated double cut inequalities;
NTC	:	number of generated triple path-cut inequalities;
NP	:	number of generated Steiner-partition inequalities;
NSP	:	number of generated Steiner-SP-partition inequalities;
COpt	:	weight of the best upper bound obtained;
Gap	:	the relative error between the best upper bound (the optimal solution if the problem has been solved to optimality) and the lower bound obtained at the root node of the Branch-and-Cut tree;
NSub	:	number of subproblems in the Branch-and-Cut tree;
TT	:	total CPU time in hours:min:sec.

The instances indicated with "*" are those for which the algorithm has not finished the computation of the root node of the Branch-and-Cut tree after the CPU time limit. The entries in the tables for these instances are given in italic. Also, for some instances, the algorithm runs out of ressources (lack memory). For these instances, we give the results we have obtained during the time the algorithm runned. These instances are indicated with "**".

The main objective of these experiments is to check the effeciency of the different formulations introduced before for solving the kHNDP. It also aims to compare each formulation with the others and compare the algorithms depending on the connectivity requirement. Obviously, we have used the same test problems with each formulation and each value of L.

Our first series of experiments concerns kHNDP$_{Ag}$ with $k = 3$ and $L = 2, 3$. The instances we have considered have graphs with 21 up to 52 nodes and a number of demands up to 50. The results are summurized in Tables 6.1 and 6.2.

$\|V\|$	$\|D\|$	NC	NAC	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
r 21	15	1963	0	0	24	0	0	7138	9.5	151	0:00:15
r 21	17	2463	2	0	25	0	0	7790	9.34	359	0:00:35
r 21	20	4076	12	0	73	0	0	8762	11.6	2195	0:06:10
a 21	10	358	51	87	0	0	0	8313	3.19	57	0:00:08
r 30	15	3482	15	0	11	0	0	12512	5.56	435	0:01:22
r 30	20	7084	138	0	31	0	0	14215	6.84	4567	0:26:55
r 30	25	8379	27	0	70	0	0	15610	8.57	3845	0:34:07
a 30	10	518	566	0	0	0	0	12124	4.96	375	0:01:16
a 30	15	862	1141	0	0	0	0	15868	3.36	1193	0:13:54
r 48	20	12780	0	0	38	0	0	21586	8.16	267	0:08:23
r 48	30	46392	0	0	5	0	0	34144	27.18	1581	5:00:00
r 48	40	42461	0	0	6	0	0	49698	37.23	1131	5:00:00
a 48	15	3514	365	2562	0	0	0	32097	2.68	891	0:28:42
a 48	20	11990	640	3754	0	0	0	46967	8.9	3993	5:00:00
a 48	24	12417	210	820	0	0	0	57865	12.59	3453	5:00:00
r 52	20	22656	19	0	2	0	0	14093	6.21	2283	0:35:50
r 52	30	67301	7	0	304	0	0	18957	16.9	3289	5:00:00
r 52	40	51484	12	0	91	0	0	24780	26.04	1703	5:00:00
r 52	50	38633	0	0	49	0	0	31541	32.36	1981	5:00:00
a 52	20	2168	1434	0	0	0	0	18480	3.24	5281	2:33:47
a 52	26	5054	780	265	0	0	0	24131	3.37	5699	5:00:00

Table 6.1: Results for Aggregated formulation with $L = 2$ and $k = 3$.

It appears from that 6.1 that for $L = 2$, 14 instances over 22 have been solved to optimality within the time limit. The CPU time for these instances, except the last one, is less than 35 minutes. All the instances of the table have required a branching phase and, for most of them, the relative error between the lower bound at the root node of the Branch-and-Cut tree and the best upper bound (Gap) is less than 10%. We also observe that our separation procedures have detected a large enough number of aggregated cut inequalities and a fewer number of double cut and triple path-cut inequalities. We observe from Table 6.2 that for $L = 3$ only 2 instances over 22 have been solved to optimality within the time limit. They have been solved respectively in 49mn and 2h34mn. Except for the previous instances, the gap between the lower

$\|V\|$	$\|D\|$	NC	NAC	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
r 21	15	23423	45	0	24	0	7	5472	8.33	975	0:49:54
r 21	17	35364	32	0	61	0	5	5864	8.24	1745	2:34:13
r 21	20	33934	5	0	58	0	0	8874	34.08	2389	5:00:00
a 21	10	51099	0	142	0	0	0	9934	38.58	347	5:00:00
a 21	11	43858	0	121	0	2	0	11390	44.6	333	5:00:00
r 30	15	55589	144	0	4	0	22	10901	13.56	2009	5:00:00
r 30	20	51627	24	0	1	0	18	15944	35.45	1835	5:00:00
r 30	25	45492	3	0	11	0	6	20379	45.53	917	5:00:00
a 30	10	39785	0	3	0	0	2	12365	21.82	1127	5:00:00
a 30	15	44901	12	43	0	0	0	23481	47.64	353	5:00:00
r 48	20	61029	0	0	11	0	19	25605	41.22	387	5:00:00
r 48	30	68969	0	0	12	0	2	40871	55.61	205	5:00:00
r 48	40	67303	0	0	0	0	1	59513	62.81	133	5:00:00
a 48	15	72110	0	22	0	0	1	62557	66.8	29	5:02:34
a 48	20	75449	0	3	0	0	0	90253	70.32	11	5:00:00
a 48	24	101539	0	3	0	0	0	121740	74.18	3	5:00:00
r 52	20	63033	0	0	0	0	15	17474	41.9	543	5:00:00
r 52	30	79985	0	0	0	0	3	23345	48.06	263	5:00:00
r 52	40	86116	0	0	0	0	4	28743	51.28	143	5:00:00
r 52	50	80976	0	0	0	0	0	37051	57.46	125	5:00:00
a 52	20	76055	0	32	0	0	2	30939	53.26	19	5:00:00
a 52	26	116481	0	20	0	0	0	51870	65.45	9	5:00:00

Table 6.2: Results for Aggregated formulation with $L = 3$ and $k = 3$.

bound at the root node of the Branch-and-Cut tree and the best upper bound is more than 10%. It even reaches in some cases 70%. We also have that our separation procedures have detected a few number of aggregated cut, double cut, triple path-cut and Steiner-SP-partition inequalities.

Our second series of experiments concerns kHNDP$_{Cu}$ with $k = 3$ and $L = 2, 3$. The results are given in Tables 6.3 and 6.4 for $L = 2$ and $L = 3$ respectively.

$\|V\|$	$\|D\|$	NC	NAC	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
r 21	15	22047	0	0	24	0	0	7138	9.5	151	1:18:44
r 21	17	42621	22208	0	6	0	0	8584	17.73	63	5:00:00
r 21	20	49283	0	0	0	0	0	10444	25.84	31	5:00:00
a 21	10	231	150	70	0	0	0	8313	3.22	71	0:00:05
a 21	11	330	163	14	0	1	0	8677	3.11	99	0:00:06
r 30	15	11437	35413	0	0	0	0	13114	9.89	43	5:00:00
r 30	20	47879	0	0	0	0	0	16488	19.68	31	5:00:00
*r 30	25	61391	0	0	0	0	0	-	-	1	5:00:00
a 30	10	450	2074	0	0	0	0	12124	4.96	359	0:02:38
a 30	15	698	2527	0	0	0	0	15868	3.33	947	0:17:20
r 48	20	34042	0	0	0	0	0	25112	21.06	27	5:00:00
*r 48	30	75649	0	0	0	0	0	-	-	1	5:00:00
*r 48	40	25240	0	0	0	0	0	-	-	1	5:00:00
a 48	15	1604	1402	830	0	0	0	32097	2.7	491	0:30:03
a 48	20	3284	3641	887	0	0	0	47449	9.95	2793	5:00:00
a 48	24	3567	2134	404	0	0	0	57308	11.48	3019	5:00:00
r 52	20	56127	0	0	0	0	0	17039	22.43	3	5:00:00
*r 52	30	38286	0	0	0	0	0	-	-	1	5:00:00
*r 52	40	24510	0	0	0	0	0	-	-	1	5:00:00
*r 52	50	24644	0	0	0	0	0	-	-	1	5:00:00
a 52	20	1474	4513	0	0	0	0	18480	3.24	3185	4:13:36
a 52	26	2656	2894	142	0	0	0	24416	4.51	3669	5:00:00

Table 6.3: Results for Cut formulation with $L = 2$ and $k = 3$.

We observe that for $L = 2$ (Table 6.3), 6 instances over 22 have been solved to optimality within the time limit. Also, for 6 instances, the algorithm has not been able to finish within 5 hours the resolution of the root node of the Branch-and-Cut tree. A large enough number of aggregated cut inequalities has been detected. However only a few number of double cut inequalities has been used. For $L = 3$ (Table 6.4), no

$\|V\|$	$\|D\|$	NC	NAC	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
r 21	15	20526	1344	0	12	0	3	7801	35.7	287	5:00:00
r 21	17	20852	95	0	1	0	1	7688	30.01	169	5:00:00
r 21	20	15636	0	0	11	0	0	10183	42.55	407	5:00:00
a 21	10	24143	0	0	0	0	0	10808	43.55	395	5:00:00
a 21	11	23988	0	1	0	1	0	9970	36.71	317	5:00:00
r 30	15	6854	0	0	0	0	7	18349	48.65	21	5:00:00
r 30	20	11332	0	0	0	0	2	21552	52.25	21	5:00:00
r 30	25	14842	0	0	0	0	0	22829	51.38	7	5:00:00
a 30	10	17955	0	0	0	0	1	12365	21.82	567	5:00:00
a 30	15	14218	66	1	0	0	0	24360	49.53	171	5:00:00
*r 48	20	2729	0	0	0	0	0	-	-	1	5:00:00
*r 48	30	3833	0	0	0	0	0	-	-	1	5:00:00
r 48	40	5772	0	0	0	0	0	67381	67.15	3	5:00:00
*a 48	15	3600	0	0	0	0	0	-	-	1	5:00:00
*a 48	20	2700	0	0	0	0	0	-	-	1	5:00:00
*a 48	24	2928	0	0	0	0	0	-	-	1	5:00:00
*r 52	20	2338	0	0	0	0	0	-	-	1	5:00:00
*r 52	30	3358	0	0	0	0	0	-	-	1	5:00:00
*r 52	40	3743	0	0	0	0	0	-	-	1	5:00:00
*r 52	50	5332	0	0	0	0	0	-	-	1	5:00:00
*a 52	20	3657	0	0	0	0	0	-	-	1	5:00:00
a 52	26	7437	0	0	0	0	0	52501	65.93	3	5:00:00

Table 6.4: Results for Cut formulation with $L = 3$ and $k = 3$.

instance has been solved to optimality within the time limit and for 9 instances over 22, the root node of the Branch-and-Cut tree has been solved after 5 hours. The gap between the lower bound at the root node of Branch-and-Cut tree and the best upper bound, when they exist, is between 30% and 50% in general. However, in some cases it reaches 67%.

The third series of experiments concerns the kHNDP$_{PA}$ with $k = 3$ and $L = 2, 3$. The results are given in Tables 6.5 for $L = 2$ and 6.6 for $L = 3$. Recall that for this formulation, we have used a Branch-and-Cut-and-Price algorithm and that the aggregated cut inequalities are not valid. Thus, they don't appear in Tables 6.5 and 6.6.

	$\|V\|$	$\|D\|$	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
	r 21	15	0	24	0	0	7138	9.5	151	0:00:10
	r 21	17	0	19	0	0	7790	9.34	309	0:01:13
	r 21	20	0	91	0	0	8762	11.6	2491	0:04:07
	a 21	10	74	0	0	0	8313	3.43	85	0:00:03
	a 21	11	14	0	0	0	8677	3.38	103	0:00:06
	r 30	15	0	3	0	0	12512	5.56	303	0:00:49
	r 30	20	0	24	0	0	14215	6.84	4731	0:28:50
**	r 30	25	0	94	0	0	15896	10.22	8226	3:12:00
	a 30	10	0	0	0	0	12124	5.2	335	0:00:31
	a 30	15	0	0	0	0	15868	3.68	943	0:02:27
	r 48	20	0	46	0	0	21586	8.16	265	0:07:17
**	r 48	30	0	100	0	0	32284	22.99	6779	4:37:00
	r 48	40	0	29	0	0	47331	34.09	7167	5:00:00
	a 48	15	0	2	0	0	17626	6.15	215	0:01:27
**	a 48	20	1762	0	0	0	46446	8.10	8599	3:57:00
**	a 48	24	776	0	0	0	55877	8.51	7583	3:52:00
	r 52	20	0	3	0	0	14093	6.21	2807	0:43:36
**	r 52	30	0	501	0	0	18497	14.84	5431	4:48:00
	r 52	40	0	207	0	0	24626	25.58	6145	5:00:00
	r 52	50	0	79	0	0	31541	32.36	3931	5:00:00
	a 52	20	0	0	0	0	18480	3.43	6547	3:02:08
	a 52	26	231	0	0	0	24125	4.11	9825	5:00:00

Table 6.5: Results for Path-Arc formulation with $L = 2$ and $k = 3$.

When $L = 2$, we can see that 13 instances over 22 have been solved to optimality within a CPU time which does not exceed 43 minutes except for the last one which

$\|V\|$	$\|D\|$	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
r 21	15	0	65	0	10	5472	8.33	867	0:36:25
r 21	17	0	92	0	9	5864	8.24	1855	1:53:37
r 21	20	0	130	0	0	8445	30.73	3627	5:00:00
a 21	10	138	0	0	0	-	-	-	3:35:00
a 21	11	38	0	1	0	6770	6.8	4155	1:46:36
r 30	15	0	45	0	23	10114	6.68	2185	5:00:00
r 30	20	0	13	0	14	15767	34.73	1553	5:00:00
r 30	25	0	21	0	5	20511	45.88	675	5:00:00
a 30	10	0	0	0	15	10254	5.73	4833	5:00:00
a 30	15	18	0	0	0	19420	36.7	2853	5:00:00
r 48	20	0	22	0	34	26721	43.68	69	5:00:00
r 48	30	0	44	0	8	40197	54.87	49	5:00:00
r 48	40	0	38	0	1	59762	62.97	19	5:00:00
a 48	15	2	0	0	0	49102	57.73	101	5:00:00
a 48	20	2	0	0	0	70272	61.88	55	5:00:00
a 48	24	2	0	0	0	85625	63.29	13	5:00:00
r 52	20	0	6	0	12	17894	43.27	83	5:00:00
r 52	30	0	12	0	4	24970	51.44	49	5:00:00
r 52	40	0	12	0	5	28530	50.92	19	5:00:00
r 52	50	0	46	0	0	38734	59.31	1	5:00:00
a 52	20	27	0	0	2	27739	47.89	45	5:00:00
a 52	26	15	0	0	0	41535	56.92	13	5:00:00

Table 6.6: Results for Path-Arc formulation with $L = 3$ and $k = 3$.

has been solved in 3 hours. For most of the instances, the gap between the lower bound at the root node of the Branch-and-Cut tree and the best upper bound is less than 32%. The separation procedures have detected a few number of double cut and triple path-cut inequalities. Also we have observed that in most cases, after the root node of the Branch-and-Cut tree, the column generation algorithm has not added new variables in the current basis. When $L = 3$, 3 instances over 22 have been solved to optimality. The CPU time used to solve them is between 36 minutes and near 2 hours. A few number of double cut, triple path-cut and Steiner-SP-partition inequalities have been detected. The gap between the best lower and upper bounds is less than 62%.

Our next series of experiments concerns the kHNDP$_{NA}$ with $k = 3$ and $L = 2$ (Table 6.7) and for $L = 3$ (Table 6.8). Here also, the aggregated cut inequalities are not valid for kHNDP$_{NA}$ and do not appear in the table.

	$\|V\|$	$\|D\|$	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
	r 21	15	0	21	0	0	7138	9.5	203	0:00:11
	r 21	17	0	19	0	0	7790	9.34	333	0:00:27
	r 21	20	0	88	0	0	8762	11.6	2621	0:03:38
	a 21	10	74	0	0	0	8313	3.43	85	0:00:02
	a 21	11	12	0	1	0	8677	3.11	107	0:00:05
	r 30	15	0	9	0	0	12512	5.56	337	0:00:54
	r 30	20	0	20	0	0	14215	6.84	4993	0:32:46
	r 30	25	0	84	0	0	15610	8.57	5087	1:07:49
	a 30	10	0	0	0	0	12124	5.2	335	0:00:26
	a 30	15	0	0	0	0	15868	3.68	947	0:02:30
	r 48	20	0	38	0	0	21586	8.16	259	0:06:37
**	r 48	30	0	0	0	0	33114	24.92	3147	3:38:00
**	r 48	40	0	0	0	0	47464	34.28	2399	4:14:00
	a 48	15	867	0	0	0	32097	2.85	351	0:08:23
**	a 48	20	1508	0	0	0	46118	7.53	4409	2:43:00
**	a 48	24	603	0	0	0	55623	9.19	3817	2:44:00
	r 52	50	0	67	0	0	31541	32.36	3149	5:00:00
	r 52	10	0	0	0	0	8299	2.35	15	0:00:02
	r 52	20	0	1	0	0	14093	6.21	1541	0:40:35
	a 52	20	0	0	0	0	18480	3.43	5969	2:41:04
**	a 52	26	193	0	0	0	24364	5.06	3231	3:19:00

Table 6.7: Results for Node-Arc formulation with $L = 2$ and $k = 3$.

From Table 6.7 we can see that, for $L = 2$, 14 instances over 22 have been solved to

	$\|V\|$	$\|D\|$	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
	r 21	12	0	0	11	7	4658	3.53	107	0:02:45
	r 21	15	0	28	0	5	5472	8.33	1033	0:29:01
	r 21	17	0	53	0	7	5864	8.24	1885	1:27:58
**	a 21	10	83	0	0	0	6886	11.40	5041	3:35:00
	a 21	11	22	0	1	0	6770	6.8	4269	1:18:48
	r 30	15	0	10	0	24	10142	7.1	2857	5:24:33
	r 30	20	0	1	0	11	16157	36.3	1377	5:09:22
	r 30	25	0	6	0	2	21330	47.96	439	5:00:00
	a 30	10	0	0	0	13	10254	5.73	4937	4:35:04
**	a 30	15	10	0	0	0	-	-	-	3:35:00
	r 48	20	0	1	0	9	27126	44.52	71	5:00:00
	r 48	30	0	0	0	0	41350	56.12	27	5:00:00
	r 48	40	0	0	0	1	60165	63.21	11	5:00:00
	a 48	15	0	0	0	0	67328	69.17	107	5:00:00
	a 48	20	0	0	0	0	86553	69.05	51	5:00:00
	a 48	24	0	0	0	0	113754	72.37	33	5:00:00
	r 52	20	0	0	0	7	19713	48.5	45	5:00:00
	r 52	30	0	0	0	0	25870	53.13	17	5:00:00
	r 52	40	0	0	0	0	28530	50.92	9	5:00:00
	r 52	50	0	0	0	0	37933	58.45	7	5:00:00
	a 52	20	13	0	0	2	27870	48.14	35	5:00:00
	a 52	26	2	0	0	0	45709	60.85	13	5:00:00

Table 6.8: Results for Node-Arc formulation with $L = 3$ and $k = 3$.

optimality. The maximum CPU time for these instances is 2h41mn and most of them are solved in less than 6 minutes. The gap between the best lower and upper bounds is, in general, less than 10%. The separation algorithms have generated a few number of double cut and triple path-cut inequalities. For $L = 3$ (Table 6.8), 7 instances have been solved to optimality. For most of the instances, the gap between the best lower and upper bounds is less than 60% but reaches in some cases 72%. We can see that a few number of double cut, triple path-cut and Steiner SP-Partition have been detected during the resolution.

When comparing, for each table, the results obtained for $L = 2$ and $L = 3$ when $k = 3$, we observe that the number of instances solved to optimality when $L = 2$ is greater than that when $L = 3$. Also the gap between the best lower and upper bounds, is in most cases, better when $L = 2$ than when $L = 3$. This let us believe that the kHNDP is easier when $L = 2$ than when $L = 3$.

Also, when comparing Tables 6.1, 6.3, 6.5 and 6.7 for $L = 2$, and Tables 6.2, 6.4, 6.6 and 6.8 for $L = 3$, we observe that the effeciency of the different algorithms for solving the problem is not the same. We observe that the results for $k\text{HNDP}_{Ag}$, $k\text{HNDP}_{PA}$ and $k\text{HNDP}_{NA}$ are better than those of $k\text{HNDP}_{Cu}$ for both $L = 2$ and $L = 3$. In fact the number of instances solved to optimality for this later formulation is less than that of the others and, in most cases, the gaps between the best lower and upper bounds are greater for this formulation than those of the other formulations. Moreover, for 6 instances for $L = 2$ and 9 instances for $L = 3$, the algorithm for $k\text{HNDP}_{Cu}$ has not been able to finish the resolution of the root node of the Branch-and-Cut tree whereas the other algorithms have solved the problem for the same instances with a branching phase. Hence, for these instances, the algorithm for $k\text{HNDP}_{Cu}$ does not produce an upper bound of the optimal solution. Comparing Tables 6.1 to 6.5 for $L = 2$, and Tables 6.2 and 6.6 for $L = 3$, we observe that the number of instances solved to optimality is quite the same for the two formulations, and the CPU times are generally closer. However, for $L = 2$, the gap between the best lower and upper bounds is, in most cases, better for the Aggregated formulation than for those of the Path-Arc formulation. Also, for $L = 3$, we notice that the gap is in general better for the Path-Arc formulation. In fact, for this latter formulation, the gap is up to 63.29% whereas it reaches 74.18% for the Aggregated formulation. The comparison between Tables 6.7 and 6.8 on the one hand and Tables 6.1, 6.2, 6.5 and 6.6 on the other hand shows that more instances have been solved to optimality by the Node-Arc formulation for both $L = 2$ and $L = 3$. The CPU time is slightly better with this formulation and the gaps between the best lower and upper bounds are better in some cases than those obtained for the Aggregated and Path-Arc formulation.

As a conclusion, these observations show that the Aggregated, Path-Arc and Node-Arc formulations are more efficient than the Cut formulation. Also, the Node-Arc formulation solves more instances to optimality while the Aggregated formulation produces better upper bounds when $L = 2$ and the Path-Arc formulation gives better ones when $L = 3$. Also, the problem is easier to solve when $L = 2$.

Our last series of experiments concerns the kHNDP with $k = 4, 5$ and $L = 3$ (Tables 6.9 and 6.10). It aims to observe the easiness of the problem when the connectivity requirement increases. The instances used have graphs with 48 and 52 nodes and up to 50 demands. Note that when $k = 4$ the Steiner-partition and Steiner-SP-partition inequalities are redundant with respect to the st-cut inequalities. Thus, they do not appear in Table 6.9.

Aggregated formulation

$\|V\|$	$\|D\|$	NC	NAC	NDC	NTC	COpt	Gap	NSub	TT
r 48	30	56483	0	0	0	48899	49.32	249	5:00:00
r 48	40	60857	0	0	0	60090	49.87	123	5:00:00
a 48	20	57931	0	1	0	112414	68.11	11	5:00:00
a 48	24	79543	0	1	0	157063	73.26	3	5:00:00
r 52	40	74438	0	0	0	34100	44.03	131	5:00:00
r 52	50	75463	0	0	0	41894	48.00	91	5:00:00
a 52	20	64736	0	32	0	39863	50.35	21	5:00:00
a 52	26	77734	0	11	0	66306	63.02	9	5:00:00

Cut formulation

$\|V\|$	$\|D\|$	NC	NAC	NDC	NTC	COpt	Gap	NSub	TT
** r 48*	*30*	*3684*	*0*	*0*	*0*	-	-	*1*	*5:00:00*
48	40	5788	0	0	0	69349	56.56	3	5:00:00
** a 48*	*20*	*2760*	*0*	*0*	*0*	-	-	*1*	*5:00:00*
** a 48*	*24*	*3408*	*0*	*0*	*0*	-	-	*1*	*5:00:00*
** r 52*	*40*	*3995*	*0*	*0*	*0*	-	-	*1*	*5:00:00*
r 52	50	6619	0	0	0	45587	52.21	3	5:00:00
** a 52*	*20*	*5832*	*0*	*0*	*0*	-	-	*1*	*5:00:00*
a 52	26	10303	0	0	0	59807	59.01	3	5:00:00

Path-Arc formulation

$\|V\|$	$\|D\|$	NDC	NTC	COpt	Gap	NSub	TT
r 48	30	0	0	49758	50.2	47	5:00:00
r 48	40	0	0	64253	53.12	19	5:00:00
a 48	20	1	0	92597	61.29	53	5:00:00
a 48	24	0	0	111039	62.18	15	5:00:00
r 52	40	0	0	32552	41.37	15	5:00:00
** r 52*	*50*	*0*	*0*	-	-	*1*	*5:00:00*
a 52	20	32	0	34525	42.67	39	5:00:00
a 52	26	9	0	54694	55.17	13	5:00:00

Node-Arc formulation

$\|V\|$	$\|D\|$	NDC	NTC	COpt	Gap	NSub	TT
r 48	30	0	0	50894	51.31	25	5:00:00
r 48	40	0	0	64495	53.29	11	5:00:00
a 48	20	0	0	111168	67.75	51	5:00:00
a 48	24	0	0	135650	69.04	31	5:00:00
r 52	40	0	0	35724	46.57	9	5:00:00
r 52	50	0	0	45536	52.16	5	5:00:00
a 52	20	1	0	39347	49.71	35	5:00:00
a 52	26	0	0	57370	57.26	13	5:00:00

Table 6.9: Results for Aggregated formulation with $L = 3$ and $k = 4$.

Aggregated formulation

$\|V\|$	$\|D\|$	NC	NAC	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
r 48	30	57487	0	0	21	0	0	54677	41.45	259	5:00:00
r 48	40	51981	0	0	13	0	0	67290	42.72	157	5:00:00
a 48	20	46889	0	0	0	0	0	140927	68.02	15	5:00:00
a 48	24	64629	0	0	0	0	0	207928	74.64	3	5:00:00
r 52	40	62674	0	0	0	0	0	38257	36.11	163	5:00:00
r 52	50	75568	0	0	9	0	0	48095	41.52	93	5:00:00
a 52	20	55999	0	28	0	0	0	46728	45.57	25	5:00:00
a 52	26	63377	0	2	0	0	0	83433	62.1	11	5:00:00

Cut formulation

$\|V\|$	$\|D\|$	NC	NAC	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
*r 48	30	3789	0	0	0	0	0	-	-	1	5:00:00
r 48	40	5073	0	0	0	0	0	76132	49.37	3	5:00:00
*a 48	20	2619	0	0	0	0	0	-	-	1	5:00:00
*a 48	24	4824	0	0	0	0	0	-	-	1	5:00:00
*r 52	40	3868	0	0	0	0	0	-	-	1	5:00:00
r 52	50	8412	0	0	0	0	0	53997	47.91	3	5:00:00
a 52	20	7292	0	0	0	0	0	47687	46.67	3	5:00:00
a 52	26	9314	0	0	0	0	0	84578	62.61	3	5:00:00

Path-Arc formulation

$\|V\|$	$\|D\|$	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
r 48	30	0	37	0	0	56700	43.54	41	5:00:00
r 48	40	0	19	0	0	70057	44.98	15	5:00:00
a 48	20	3	0	0	0	106719	57.77	49	5:00:00
a 48	24	2	0	0	0	130029	59.45	7	5:00:00
r 52	40	0	3	0	0	39933	38.79	15	5:00:00
*r 52	50	0	12	0	0	-	-	1	5:00:00
a 52	20	8	0	0	0	42615	40.33	39	5:00:00
a 52	26	5	0	0	0	63315	50.06	9	5:00:00

Node-Arc formulation

$\|V\|$	$\|D\|$	NDC	NTC	NP	NSP	COpt	Gap	NSub	TT
r 48	30	0	3	0	0	58514	45.29	25	5:00:00
r 48	40	0	0	0	0	72125	46.56	13	5:00:00
a 48	20	0	0	0	0	133820	66.32	47	5:00:00
a 48	24	0	0	0	0	170278	69.03	33	5:00:00
r 52	40	0	0	0	0	40081	39.02	9	5:00:00
r 52	50	0	0	0	0	53997	47.91	5	5:00:00
a 52	20	0	0	0	0	46318	45.09	35	5:00:00
a 52	26	0	0	0	0	70195	54.96	15	5:00:00

Table 6.10: Results for Aggregated formulation with $L = 3$ and $k = 5$.

First, we remark that for $k = 4$ and $k = 5$, the instances in Tables 6.9 and 6.10 have not been solved to optimality after 5 hours. The Cut formulation has not been able to solve after 5 hours the linear relaxation of the problem at the root node of the Branch-and-Cut tree for 5 (resp. 4) instances when $k = 4$ (resp. $k = 5$).

We notice that for the Aggregated formulation, the gap between the best lower and upper bound is better when $k = 4$ than when $k = 5$. For example, when $k = 4$, the gaps are between 44.03% and 73.26% while for $k = 5$ the gaps are between 36.11% and 74.64%. Also, except one instance, the gap is better when $k = 4$ than when $k = 5$. This shows that the kHNDP is easier when $k = 4$ than when $k = 5$. The same observation can be done for the other formulations. In particular, for the Cut formulation, we see that the instance *r 52* with $|D| = 20$ has not reached the branching phase for $k = 4$ while 3 nodes have been generated in the Branch-and-Cut tree for $k = 5$. Moreover, for $k = 4$ the primal heuristic does not produce a feasible, and hence no upper bound for the optimal solution, while for $k = 5$ the algorithm produces an upper bound and a gap of 46.67%.

Also these results can be compared to those obtained for $k = 3$ and $L = 3$. We can remark that, for every formulation, the gaps between the best lower and upper bounds are better when $k = 4, 5$ than when $k = 3$. From these observations, we conjecture that the kHNDP becomes easier when the connectivity requirement k increases.

6.5 Concluding remarks

In this chapter, we have studied the k-edge-connected hop-constrained network design problem when $k \geq 3$ and $L = 2, 3$. We have presented four integer programming formulations based on the transformation of the initial graph into appropriated directed graphs. We have also introduced some classes of valid inequalities and given conditions under which these inequalities define facet of the associated polytope. We have also discussed separation procedures for these inequalities and a column generation algorithm. Using these results, we have devised Branch-and-Cut and Branch-and-Cut-and-Price algorithms to solve the problem.

The computational results have shown that the Aggregated, Path-Arc and Node-Arc formulations are effective in solving the problem and producing good upper bound for the problem and that the Cut formulation is less efficient. Also, it has been shown that the Node-Arc formulation is more efficient in solving the problem to optimality and that Aggregated and Path-Arc formulation produces good upper bound when $L = 2$ and when $L = 3$, respectively.

Also our heuristics to separate the aggregated, double cut and triple path-cut inequalities have appeared to be very efficient.

These experiments showed that the kHNDP is easier when $L = 2$ than when $L = 3$. It also showed that the problem becomes easier when the connectivity requirement increases.

In some cases, we may consider that $L \geq 4$. Few works have been done for this case in the literature. In particular, Huygens and Mahjoub [73] studied this case and showed that st-cut inequalities (5.1) and L-st-path-cut inequalities (5.2) toghether with integrality constraints are no more sufficient to formulate the problem as an integer program. They [73] introduced new classes of inequalities and showed that these inequalities toghether with integrity constraints and inequalities (5.1) and (5.2) formulate the problem in the space of the design variables. One can try to extend the approach developed in the previous chapters to study the problem when $L \geq 4$ and devise efficient Branch-and-Cut or Branch-and-Cut-and-Price algorithms for the problem in this case.

Conclusion

In this thesis, we have studied, within a polyhedral context, two survivable network design problems, the k-edge-connected subgraph (kECSP) and the k-edge-connected hop-constrained network design (kHNDP) problems. In particular, we have considered these problems in the case where a high level of connectivity is required, that is when $k \geq 3$. These two problems are NP-hard when $k \geq 2$.

First, we have discussed the polytope of the kECSP. We have introduced a new class of valid inequalities and given conditions for these inequalities to be facet defining. We have also studied further valid inequalities and given conditions under which they define facets. Moreover, we have studied the reduction operations introduced by Didi Biha and Mahjoub [39] (see also [38]). These allow to perform the separation of the valid inequalities in a reduced graph. Using these results, we have devised a Branch-and-Cut algorithm for the problem and given computational results for $k = 3, 4, 5$.

We have also studied the kHNDP when $k \geq 3$ and $L \in \{2, 3\}$. We have first investigated the problem when a single demand is considered and shown that the associated polytope is completely described by the st-cut and L-path-cut inequalities toghether with the trivial inequalities. We showed that this complete description yields a polynomial cutting plane algorithm for the problem, generalizing at the same time the results of Huygens et al. [75] and Dahl et al. [35].

Finally, we have considered the kHNDP when more than one demand are considered. We have introduced four new integer programming formulations for the problem in this case. These formulations rely on the transformation of the initial undirected graph G into appropriate directed graphs and the equivalence between edge-disjoint L-st-paths in G and arc-disjoint paths in these directed graphs. We have introduced several classes of valid inequalities for the polytopes associated with each formulation and studied conditions under which these inequalities define facets. Using this, we have devised Branch-and-Cut and Branch-and-Cut-and-Price algorithms for the problem. Computational results have been given for $k = 3, 4, 5$ and $L = 2, 3$, and a comparative

study has been done in order to compare the efficiency of the different formulations we have introduced.

The experimental studies presented throughout this thesis have shown that the two problems are easier to solve when the connectivity requirement k increases. It also appeared that the problems are more difficult to solve when k is odd. Our experiments for the kECSP also showed that reduction operations, when properly designed and implemented, can significantly improve a Branch-and-Cut algorithm. It would be interesting to extend the use of such operations for other combinatorial optimization problems.

The experiments we have performed for the kHNDP for $k = 3, 4, 5$ and $L = 2, 3$ gave gaps (relative error between the best lower and upper bounds) relativety high, in particular when a large number of demand is considered. It would be interesting to pursue the approach used here for the kHNDP when $L \in \{2, 3\}$. One may lead a deeper investigation of the polytope of the problem by using the appropriate directed graphs and exploiting the known results on arc-disjoint paths problems in directed graphs. This may help to provide new facet defining inequalities. It would also be interesting, from an algorithmic point of view, to improve the separation procedures provided for the various inequalities we have introduced in this work, especially for the aggregated cut inequalities.

The same kind of study can also be used for the kHNDP when $L \geq 4$. If possible, this may provide an integer programming formulation for the problem as well as a Branch-and-Cut algorithm for all $L \geq 4$ and $k \geq 2$.

Bibliography

[1] ABACUS - A Branch-And CUt System, "http://www.informatik.uni-koeln.de/abacus".

[2] Cplex, "http://www.ilog.com".

[3] TSPLIB, "http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/".

[4] D. Applegate, R. Bixby, V. Chvatal and W. Cook, "Implementing the Dantzig-Fulkerson-Johnson algorithm for large travelling saleman problems", *Mathematical Programming 97*, 2003, pp. 91-153.

[5] M. Baiou, "Le probléme du sous-graphe Steiner 2-arête-connexe: approche polyédrale", PhD. Thesis, Université Rennes 1, 1996.

[6] M. Baïou, F. Barahona and A. R. Mahjoub, "Separation of partition inequalities", *Mathematics of Operations Research 25*, 2000, pp. 243-254.

[7] M. Baïou and A. R. Mahjoub, "Steiner 2-edge connected subgraph polytopes on series-parallel graphs", *SIAM Journal on Discrete Mathematics 10*, 1997, pp. 505-514.

[8] F. Barahona and A. R. Mahjoub, "On two-connected subgraph polytopes", *Discrete Mathematics 147*, 1995, pp. 19-34.

[9] C. Barhart, C. A. Hane, P. H. Vance, "Using Branch-and-Price-and-Cut to Solve Origin-Destination Integer Multicommodity Flow Problems", *Operations Research 48*, 2000, pp. 318-326.

[10] C. Barhart, E. L. Johnson, G. L. Nemhauser, G. L. Savelsberg and P. H. Vance, "Branch-and-Price: Column generation for solving huge integer programs", *Operations Research 46*, 1998, pp. 316-329.

[11] R. E. Bellman, "On a Routing Problem", *Quarterly of Applyied Mathematics 16 (1)*, pp. 87-90, 1958.

[12] F. Bendali, I. Diarrassouba, M. Didi Biha, A. R. Mahjoub and J. Mailfert, "A Branch-and-Cut algorithm for the k-edge connected subgraph problem", *to appear in Networks*.

[13] F. Bendali, I. Diarrassouba, A. R. Mahjoub and J. Mailfert, "The k edge-disjoint 3-hop-constrained paths polytope", *submitted to Discrete Optimization*.

[14] D. Bienstock, E. F. Brickel and C. L. Monma, "On the structure of minimum weight k-connected networks", *SIAM Journal on Discrete Mathematics 3*, 1990, pp. 320-329.

[15] J. A. Bondy and U. S. R. Murty, "Graph Theory with Applications", *University Press, Belfast*, 1976.

[16] S. Borne, "Sécurisation et dimensionnement de réseaux multicouches: modèles et polyêdres", *PhD. Thesis*, Université Blaise Pascal, Clermont-Ferrand II, 2006.

[17] S. C. Boyd and T. Hao, "An integer polytope related to the design of survivable communication network", *SIAM Journal on Discrete Mathematics 6*, 1993, pp. 612-630.

[18] G. R. Cai and Y. G. Sun, "The minimum augmentation of any graph to a k-edge-connected graph", *Networks 19*, 1989, 151-172.

[19] J. Cheriyan, A. Sebö and Z. Sziget, "An improved approximation algorithm for minimum size 2-edge connected subgraphs", *Proceedings of the* 6^{th} *International IPCO (Integer Programming and Combinatorial Optimization) Conference*, 1998, pp. 126-136.

[20] J. Cheriyan and R. Thurimella, "Approximating Minimum-Size k-Connected Spanning Subgraphs via Matching", *SIAM Journal on Computing 30 (2)*, 1998, pp. 292-301.

[21] S. Chopra, "The k-edge connected spanning subgraph polyhedron", *SIAM Journal on Discrete Matematics 7*, 1994, pp. 245-259.

[22] S. A. Cook, "The complexity of theorem-proving procedures", *In procedings* 3^{rd} *Annual ACM Symposium on Theory of Computing*, New York, 1971, pp. 151-158.

[23] G. Cornuéjols, J. Fonlupt and D. Naddef, "The traveling salesman problem on a graph and some related polyhedra", *Mathematical Programming 33*, 1985, pp. 1-27.

[24] C. R. Coullard, A. B. Gamble and J. Lui, "The k-walk polyhedron", *Advances in Optimization and Approximation, Nonconvex Optimization Application 1, D-Z Du and J. Sun editions, Kluwer Academic Publishers, Dordrecht, The Netherlands*, 1994, pp. 9-29.

[25] R. Coullard, A. Rais, R. L. Radin, D. K. Wagner, "Linear time algorithm for the 2-connected Steiner subgraph problem on special classes of graphs", *Networks 23*, 1993, pp. 195-206.

[26] R. Coullard, A. Rais, R. L. Radin, D. K. Wagner, "The Dominant of the 2-connected Steiner subgraph polytope for W_4-free graphs", *Discrete Applied Mathematics 66*, 1996, pp. 33-43.

[27] G. Dahl, "Contributions to the design of directed survivable networks", *PhD. Thesis*, University of Oslo, 1991.

[28] G. Dahl, "Directed Steiner problems with connectivity constraints", *Discrete Appilied Mathematics 47*, 1993, pp. 109-128.

[29] G. Dahl, "The Design of Survivable Directed Networks", *Telecommunication Systems 2*, 1992, pp. 349-377.

[30] G. Dahl, "The 2-hop spanning tree problem", *Operations Research Letters 23 (1-2)*, 1999, pp. 21-26.

[31] G. Dahl, "Notes on polyhedra associated with hop-constrained paths", *Operations Research Letters 25 (2)*, 1999, pp. 97-100.

[32] G. Dahl and L. Gouveia, "On the directed hop-constrained shortest path problem", *Operations Research Letters 32*, 2004, pp. 15-22.

[33] G. Dahl, N. Foldnes and L. Gouveia, "A note on hop-constrained walk polytopes". *to appear in Operations Research Letters.*

[34] G. Dahl and B. Johannessen, "The 2-path network problem", *Networks 43 (3)*, 2004, pp. 190-199.

[35] G. Dahl, D. Huygens, A. R. Mahjoub and P. Pesneau, "On the k edge-disjoint 2-hop-constrained paths polytope", *Operation Research Letters 34 (5)*, 2006, pp. 577-582.

[36] G. Dantzig and P. Wolfe, "Decomposition principle for linear programming", *Operations Research 8*, 1960, pp. 101-111.

[37] I. Diarrassouba and L. Slama, "Les inégalités de SP-partition pour le problème du sous-graphe k-arête connexe", *Research Report, LIMOS RR-07-13*, 2007.

[38] M. Didi Biha, "Graphes k-arêtes connexes et polyèdres", *PhD. Thesis*, Université de Bretagne Occidentale, Brest, 1998.

[39] M. Didi Biha and A. R. Mahjoub, "The k-edge connected subgraph problem I: Polytopes and critical extreme points", *Linear Algebra and its Applications 381*, 2004, pp. 117-139.

[40] M. Didi Biha and A. R. Mahjoub, "k-edge connected polyhedra on series-parallel graphs", *Operations Research Letters 19*, 1996, pp. 71-78.

[41] M. Didi Biha and A. R. Mahjoub, "Steiner k-edge connected subgraph polyhedra", *Journal of Combinatorial Optimization 4*, 2000, pp. 131-134.

[42] M. Didi Biha, A. R. Mahjoub and L. Slama, "On the separation of partition inequalities", *Proceedings INOC 2005*, pp. B2.500-B2.505.

[43] E. W. Dijkstra, "A note on two problems in connexion with graphs", *Numerische Mathematik. 1*, 1959, pp. 269-271.

[44] J. Edmonds, "Covers and packings in a family of sets", *Bull. American Mathematical Society 68*, 1962, pp. 494-499.

[45] J. Edmonds, "Maximum matching and a polyhedron with 0,1-vertices", *Journal of Research of the National Bureau of Standards (B) 69*, 1965, pp. 9-14.

[46] D. Eppstein, "Finding the k Shortest Paths", *SIAM Journal on Computing 28(2)*, 1999, pp. 652-673

[47] K. P. Eswaran and R. E. Tarjan, "Augmentation problems", *SIAM Journal on Computing 5*, 1976, pp. 653-665.

[48] C. G. Fernandes, "A Better Approximation Ratio for the Minimum Size k-Edge-Connected Spanning Subgraph Problem", *Journal of Algorithms 28*, 1998, pp. 105-124.

[49] J. Fonlupt and A. R. Mahjoub, "Critical extreme points of the 2-edge connected spanning subgraph polytope", *Mathematical Programming 105*, 2006, pp. 289-310.

[50] J. Fonlupt and D. Naddef, "The traveling salesman problem in graphs with some excluded minors", *Mathematical Programming 53*, 1992, pp. 147-172.

[51] B. Fortz, A. R. Mahjoub, S. T. McCormick and P. Pesneau, "Two-edge connected subgraphs with bounded rings: Polyhedral results and Branch-and-Cut", *Mathematical Programming 105 (1)*, 2006, pp. 85-111.

[52] B. Fortz, M. Labbe and F. Maffioli, "Solving the two-connected network with bounded meshes problem", *Operations Research Letters 48*, 2000, pp. 866-877.

[53] A. Frank, " Augmenting graphs to meet edge-connectivity requirements", *SIAM Journal on Discrete Mathematics 5*, 1992, pp. 22-53.

[54] G. N. Frederickson and J. Jàjà, "On the relationship between the biconnectivity augmentations and traveling salesman problem", *Theoretical Computer Science 13*, 1982, pp. 189-201.

[55] H. N. Gabow, M. X. Goemans, E. Tardos and D. P. Williamson, "Approximating the smallest k-edge connected spanning subgraph by LP-rounding", *Networks 53 (4)*, 2009, pp. 345-357.

[56] M. R. Garey and D. J. Johnson, "Computer and Intractability: A Guide to the Theory of *NP*-completness", *Freeman, San Francisco*, 1979.

[57] M. X. Goemans and D. J. Bertsimas, "Survivable network, linear programming and the parsimonious property", *Mathematical Programming 60*, 1993, pp. 145-166.

[58] A. Goldberg and R. E. Tarjan, "A New Approach to the Maximum Flow Problem", *Journal of the Association for Computing Machinery 35*, 1988, pp. 921-940.

[59] R. E. Gomory and T. C. Hu, "Multi-Terminal Network Flows", *Journal of the Society for Industrial and Applied Mathematics 9*, 1961, pp. 551-570.

[60] L. Gouveia, "Multicommodity flow models for spanning trees with hop constraints", *Operations Research Letters 25 (2)*, 1999, pp. 97-100.

[61] L. Gouveia, "Using variable redefinition for computing lower bounds for minimum spanning", *INFORMS Journal on Computing* 10 (2), 1998, pp. 180-188.

[62] L. Gouveia and P. icio and A. de Sousa and R. Valadas, "MPLS over WDM Network Design with Packet Level QoS Constraints based on ILP Models", *In Proceedings of INFOCOM 2003*, 2003.

[63] L. Gouveia and C. Requejo, "A new Lagrangean relaxation approach for the hop-contrained minimum spanning tree problem", *European Journal of Operations Research 25 (2)*, 2001, pp. 539-552.

[64] M. Grötschel, L. Lovász, A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization", *Combinatorica 1*, 1981, pp. 70-89.

[65] M. Grötschel and C. L. Monma, "Integer polyhedra arising from certain network design problems with connectivity constraints", *SIAM Journal on Discrete Mathematics 3*, 1990, pp. 502-523.

[66] M. Grötschel, C. L. Monma and M. Stoer, "Polyhedral approches to network survivability", In F. Roberts, F. Hwang and C. L. Monma, eds, *Reliability of computer and Communication newtorks, Vol 5, Series Discrete Mathematics and Computer Science, AMS/ACM*, 1991, pp. 121-141.

[67] M. Grötschel, C. L. Monma and M. Stoer, "Polyhedral and computational investigations arising for designing communication networks with high survivability requirements",*Operation Research 43*, 1995, pp. 1012-1024.

[68] D. Gusfield, "Very simple method for all pairs network flow analysis", *Society of Industrial and Applied Mathematics 009*, 1990, pp. 143-155.

[69] J. Hao and J. B. Orlin, "A faster algorithm for finding the minimum cut in a directed graph", *Journal of Algorithms*, 1992, pp. 424-446.

[70] J. Hershberger, M. Maxel, S. Suri , "ACM Transactions on Algorithms", *Transactions on Algorithms (TALG) vol 3(4)*, 2007

[71] T.-S. Hsu and M.-Y. Kao, "A unifying augmentation algorithm for the two-edge connectivity and biconnectivity", *Journal of Combinatorial Optimization 2*, 1981, pp. 237-256.

[72] D. Huygens, "Design of Survivable Networks with Bounded-Length Paths", *PhD. Thesis*, Université Libre de Bruxelles, Bruxelles, 2005.

[73] D. Huygens and A. R. Mahjoub, "Integer programming formulation for the two 4-hop-constrained paths problem", *Networks 49 (2)*, 2007, pp. 135-144.

[74] D. Huygens, A. R. Mahjoub, M. Labbe and P. Pesneau, "The two-edge connected hop-constrained network design problem: Valid inequalities and Branch-and-Cut", *Networks 49 (1)*, 2007, pp. 116-133.

[75] D. Huygens, A. R. Mahjoub and P. Pesneau, "Two edge-disjoint hop-constrained paths and polyhedra", *SIAM Journal on Discrete Mathematics 18 (2)*, 2004, pp. 287-312.

[76] D. Huygens, M. Labbe, A. R. Mahjoub and P. Pesneau, "The two edge-connected hop-constrained network design problem: valid inequalities and Branch-and-Cut", *Networks 49 (1)*, 2007, pp. 116-133.

[77] R. M. Karp, "Reducibility among combinatorial problems", *In R. E. Miller and J. W. Tatcher, editors, Complexity of Computer Computation*, 1972, pp. 85-103.

[78] D. R. Karger, "Random sampling in cut, flow and network design problems", *Mathematics of Operations Research 24*, 1999, pp. 383-413.

[79] H. Kerivin, "Réseaux fiables et polyêdres", *PhD. Thesis*, Université Blaise Pascal, Clermont-Ferrand II, 2000.

[80] H. Kerivin and A. R. Mahjoub, "Design of Survivable Networks: A Survey", *Networks 46*, 2005, pp. 1-21.

[81] H. Kerivin, A. R. Mahjoub and C. Nocq, "(1,2)-survivable networks: Facets and Branch-and-Cut", *The Sharpest Cut, MPS-SIAM Series in Optimization*, M. Grötschel (Editor), 2004, pp. 121-152.

[82] S. Khuller and B. Raghavachari, "Improved Approximation Algorithms for Uniform Connectivity Problems", *Journal of Algorithms 21*, 1996, pp. 434-450.

[83] C.-W. Ko and C. L. Monma, "Heuristics for designing highly survivable communication networks", *Technical Report, Bellcore, Morristown, New Jersey*, 1989.

[84] J. B. Kruskal, "On the shortest spanning subtree of a graph and the travelling salesman problem", *Proceedings of the American Mathematical Society 7*, 1956, pp. 48-50.

[85] I. Loiseau, A. Ceselli, N. Maculan and M. Salani, "Génération de colonnes en programmation linéaire en nombre entiers", In V. Th. Paschos, editor, *Optimisation combinatoire: concepts fondamentaux*, Chapitre 8, Hermès, Paris, 2005.

[86] M. E. Lubbecke and J. Desrosiers, "Selected topics in Column Generation", *Operation Research 53*, 2005, pp. 1007-1023.

[87] W. Mader, "A reduction method for edge-connectivity in graphs", *Annals of Discrete Mathematics 3*, 1978, pp. 145-164.

[88] A. R. Mahjoub, "Two-edge Connected spanning subgraphs and polyhedra", *Mathematical Programming 64*, 1994, pp. 199-208.

[89] A. R. Mahjoub, "On perfectly Two-edge connected subgraphs and polyhedra", *Discrete Mathematics 170*, 1997, pp. 153-172.

[90] A. R. Mahjoub, "Approches polyhédrales", In V. Th. Paschos, editor, *Optimisation combinatoire: concepts fondamentaux*, Chapitre 9, Hermès, Paris, 2005.

[91] K. Menger, "Zur allgemeinen kurventhorie", *Fundamanta Mathematicae 10*, 1927, pp. 96-115.

[92] C. L. Monma, B. S. Munson and W. R. Pulleyblank, "Minimum weight two-connected spanning networks", *Operations Research 37*, 1989, pp. 153-171.

[93] G. L. Nemhauser and L. A. Wolsey, "Integer and Combinatorial Optimization", Wiley Editions, 1988, pp. 259-295.

[94] P. Pesneau, "Conception de réseaux 2-connexes avec contraintes de bornes", *PhD. Thesis*, Université Blaise Pascal, Clermont-Ferrand II, 2003.

[95] R. C. Prim, "Shortest connection networks and some generalization", *Bell System technical Journal 36*, 1957, pp. 1389-1401.

[96] A. Schrijver, "Combinatorial Optimization - Polyhedra and Efficiency. Algorithms and Combinatorics", *Vol. 24. Springer, Berlin, Heidelberg*, 2003.

[97] L. Slama, "Conception de réseaux fiables: Séparation et Polyêdres", *PhD. Thesis*, Université Blaise Pascal, Clermont-Ferrand II, 2008.

[98] T. Soneoka, H. Nakada and M. Imase, "Design of a d-connected digraph with a minimum number of edges and a quasiminimal diameter", *Discrete Applied Mathematics 27*, 1990, pp. 225-265.

[99] M. Stoer, "Design of Survivable Networks", *PhD. Thesis*, University of Augsburg, 1992.

[100] J. W. Suurballe, "Disjoint paths in networks", *Networks 14*, 1974, pp. 125-145.

[101] S. Thienel, "ABACUS - A Branch-And-CUt System", *PhD thesis*, Universität zu Köln, 1995.

[102] F. Vanderbeck, "Decomposition and Column Generation for Integer Programming", *PhD. thesis*, Université Catholique de Louvain, Louvain, Belgique, 1994.

[103] T. Watanabe and A. Nakamura, "A minimum 3-connectivity augmentation of graph", *Journal of Computing and System Sciences 46 (1)*, 1993, pp. 91-128.

[104] P. Winter, "Generalized Steiner problem in Halin Networks", *Proceedings of* 12^{th} *International Symposium on Mathematical Programming*, MIT, 1985.

[105] P. Winter, "Generalized Steiner problem in series-parallel Networks", *Journal of Algorithm 7*, 1986, pp. 549-566.